Frameworx Specification

Onboarding Management API REST Specification

TMF650
Release 16.0.0
April 2016

Latest Update: Frameworx Release 16	Member Evaluation
Version 1.0.1	IPR Mode: RAND

NOTICE

Direct inquiries to the TM Forum office:

240 Headquarters Plaza,
East Tower – 10th Floor,
Morristown, NJ 07960 USA
Tel No. +1 973 944 5100
Fax No. +1 973 944 5110
TM Forum Web Page: www.tmforum.org

TABLE OF CONTENTS

NOTICE 2
Table of Contents 3
List of Tables 5
Introduction 6
SAMPLE USE CASES 7
RESOURCE MODEL 17
Managed Entity and Task Resource Models 17
Partnership Type resource 17
Party Role resource 19
Notification Resource Models 25
Partnership Type Creation Notification 26
Partnership Type Remove Notification 27
Party Role Creation Notification 27
Party Role Attribute Value Change Notification 27
Party Role State Change Notification 28
Party Role Remove Notification 28
API OPERATIONS 29
Operations on Partnership Type 29
List partnership types 29
Retrieve partnership type 31
Create partnership type 32
Patch partnership type 33
Delete partnership type 35
Operations on Party Role 35
List party roles 35
Retrieve party role 36
Create party role 38

Patch party role 40
Delete party role 42
API NOTIFICATIONS 44
Register listener 44
Unregister listener 45
Publish Event to listener 45
Acknowledgments 47
Version History 47
Release History 47
Contributors to Document 47

LIST OF TABLES

N/A

INTRODUCTION

The Onboarding API provides standardized mechanisms for managing an onboarding process.

The following resources are managed by this API:

- **PartnershipType**: Identifies a type of a partnership between parties, including the list of roles types that are permitted (i.e Buyer, Seller, Developper). Role types may refer to agreement specifications to be signed by parties playing the role.

- **PartyRole**: An instantiation of one of the party roles defined for a partnership type. Represents the fact that a given *party* will play a given identified role type in the context of a partnership type.

The API allows the retrieval, creation, update and deletion of partnership type and its owned sub-resources.

API Dependencies

This API has strong dependencies with the following management APIs:

- Party Management API: used to query, create, update or delete information on *individuals* or *organizations* that will be onboarded.
- Agreement Management API: used to query, create, update or delete agreements and agreement specifications. These agreements need to be created and updated when signed by the involved parties.
- Account Management API: used to retrieve, create, update or delete differents kinds of accounts that made be needed in the context of the onboarding process, such as billing or settlement accounts and financial accounts.

Other indirect APIs dependencies when using this API are:

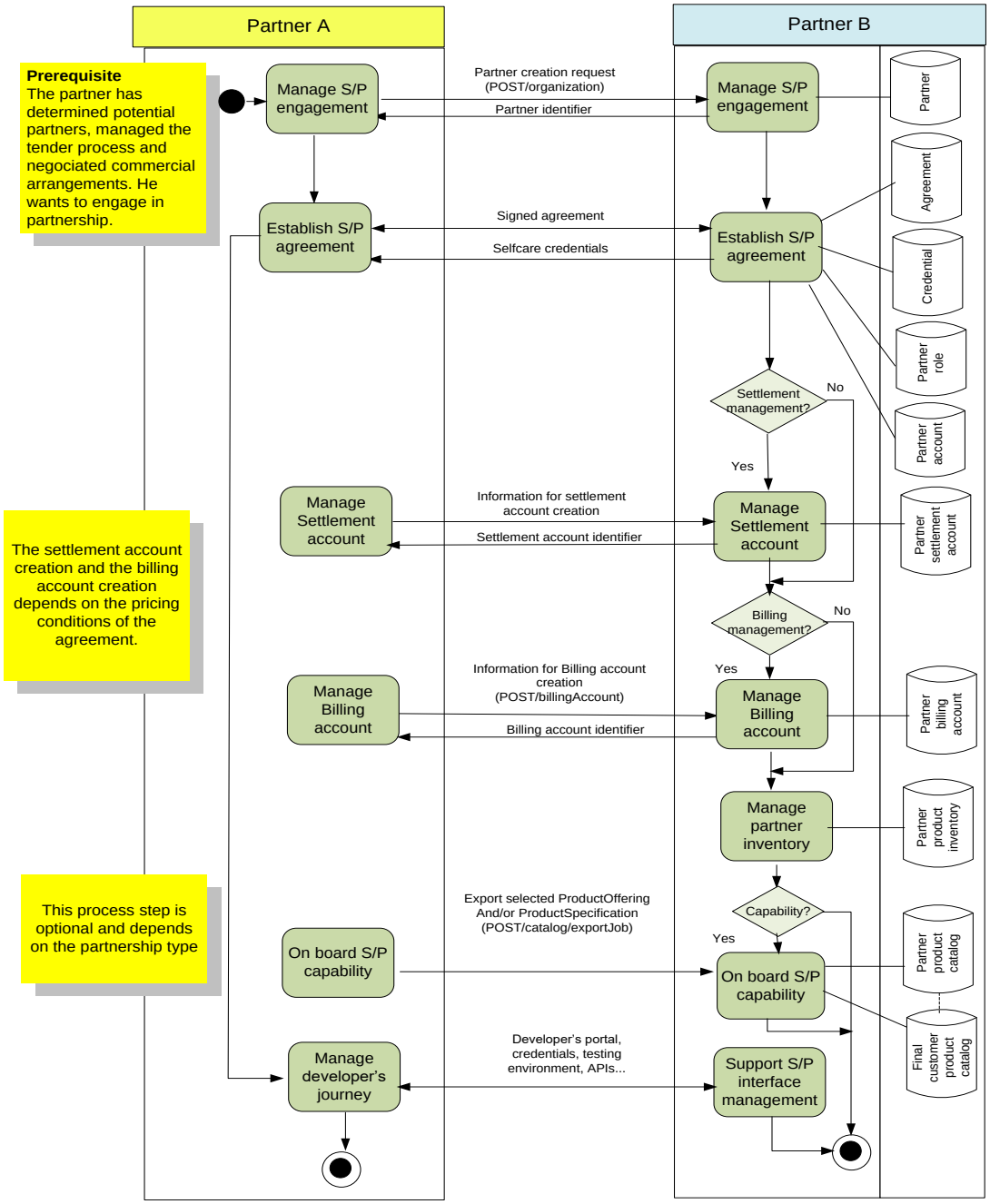
- Product Catalog Management API: used to connect agreements to product offerings
- Product Inventory Management API: retrieval of products related to product offerings.
- Product Ordering management API: establishing order on available products.

SAMPLE USE CASES

In this section we provide some typical API usage scenarios described in a light-weight fashion. The intent is not to describe all possible contexts of use of the API. Much more details on these use cases can be found in “Open Digital Business Scenarios and Use Cases” document.

A Global view

The figure below depicts the on-boarding business process at a very high level of abstraction.



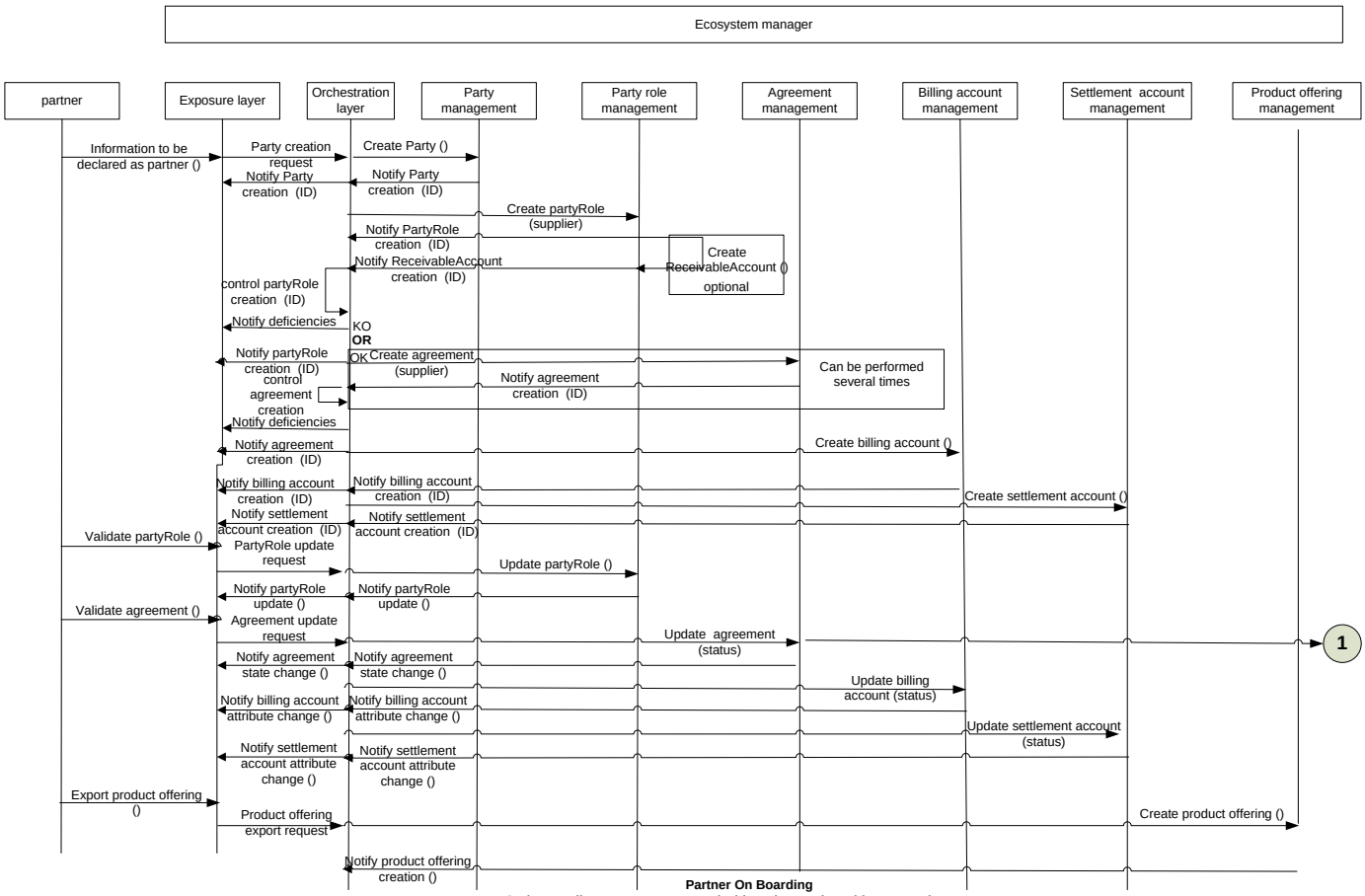
Usage options for the API

In the following we provide a list of sequence diagrams illustrating different usages of the API. A partner starts the interaction by accessing an “exposure layer”, which in turn communicates with an “orchestration layer” which make calls to the On Boarding API.

Note: In these diagrams we assume that the “orchestration layer” performs direct calls to the APIs and implements as well the “hub” client listener interfaces to receive the notifications from the API. In contrast the way how the “orchestration layer” communicates with the “exposure layer” could be done in any way and does not assume for the “exposure layer” to support the API notification listener interfaces.

Option 1: all resources are created without interaction with prospective partner

(Sequence diagram corresponding to on boarding context details document description)



This sequence diagram describes a partner on boarding process where:

- Prospective partner is not known by ecosystem manager system
- All resources (party, partyRole, agreement, billing account and / or settlement account) are created without interaction with prospective partner
- Receivable account is created optionally
- Business logic:
    - Controls party role creation and notifies exposure layer of partyRole creation or deficiencies if there are errors or inconsistencies

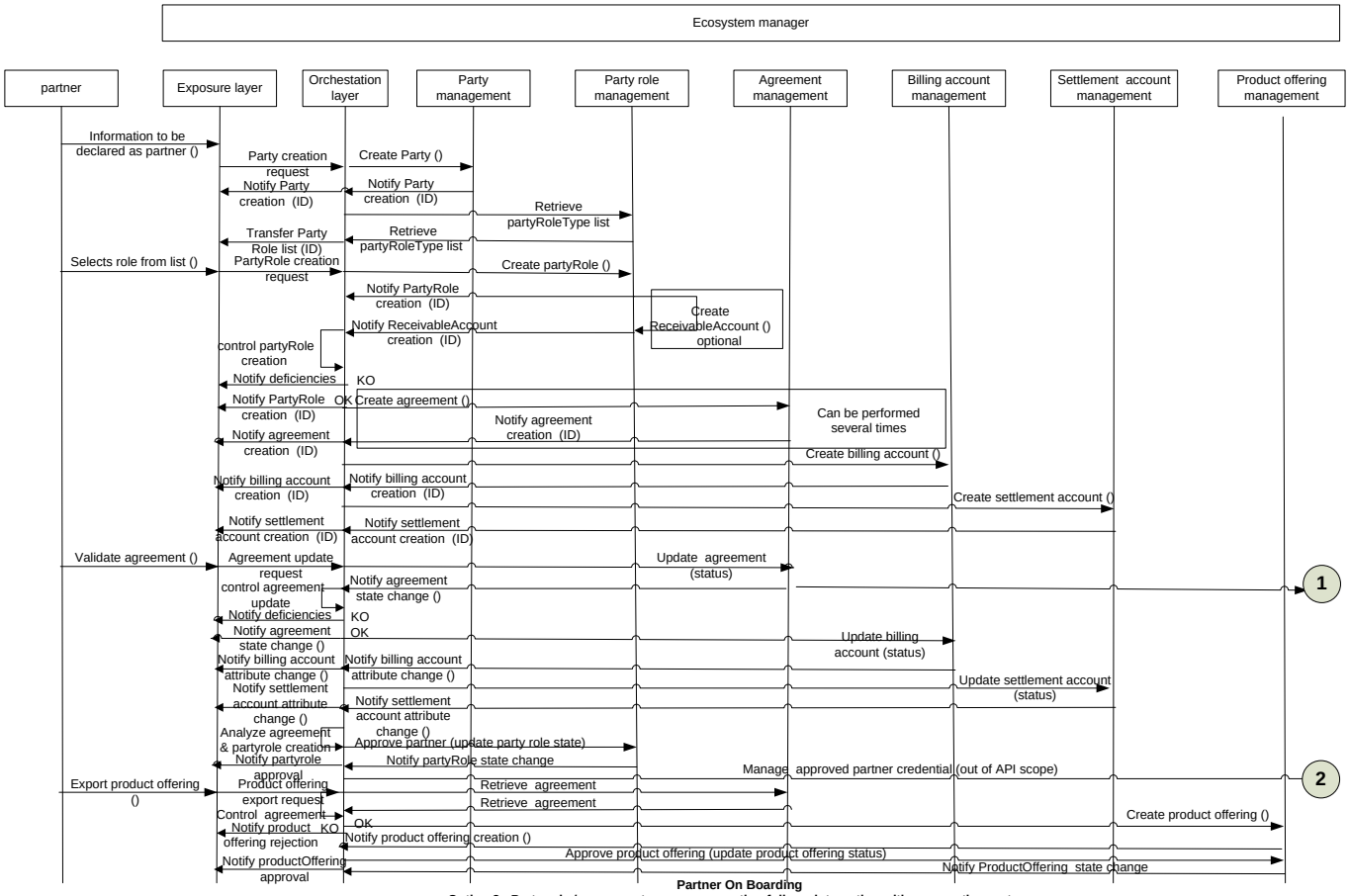
- Controls agreement creation and notifies exposure layer of agreement creation or deficiencies if there is errors or inconsistencies
- Agreement creation can be performed several times
- Prospective partner validate partyRole and, then, partyRole is updated
- Prospective partner validate agreement and, then, agreement, billing account and / or settlement account are updated
- Business logic manage partner credentials (out of API scope)
- Partner creates a product offering if allowed

Note:

- For the sake of readability, following actions are not presented on the sequence diagram
  - Subsequent actions in case of deficiencies when controlling partyRole or Agreement creation,
  - Partner approval when it depends on partyRole and agreement creation
  - Verification if partner productOffering creation request is covered by an agreement
  - Product offering approval or rejection by ecosystem manager.

Option 2: Party role / agreement resources creation follows interaction with prospective partner

(Agreement is selected automatically based on partyrole chosen)



Partner On Boarding

Option 2 : Party role / agreement resources creation follows interaction with prospective partner
(agreement is selected automatically based on partyrole chosen)

This sequence diagram describes a partner on boarding process where:

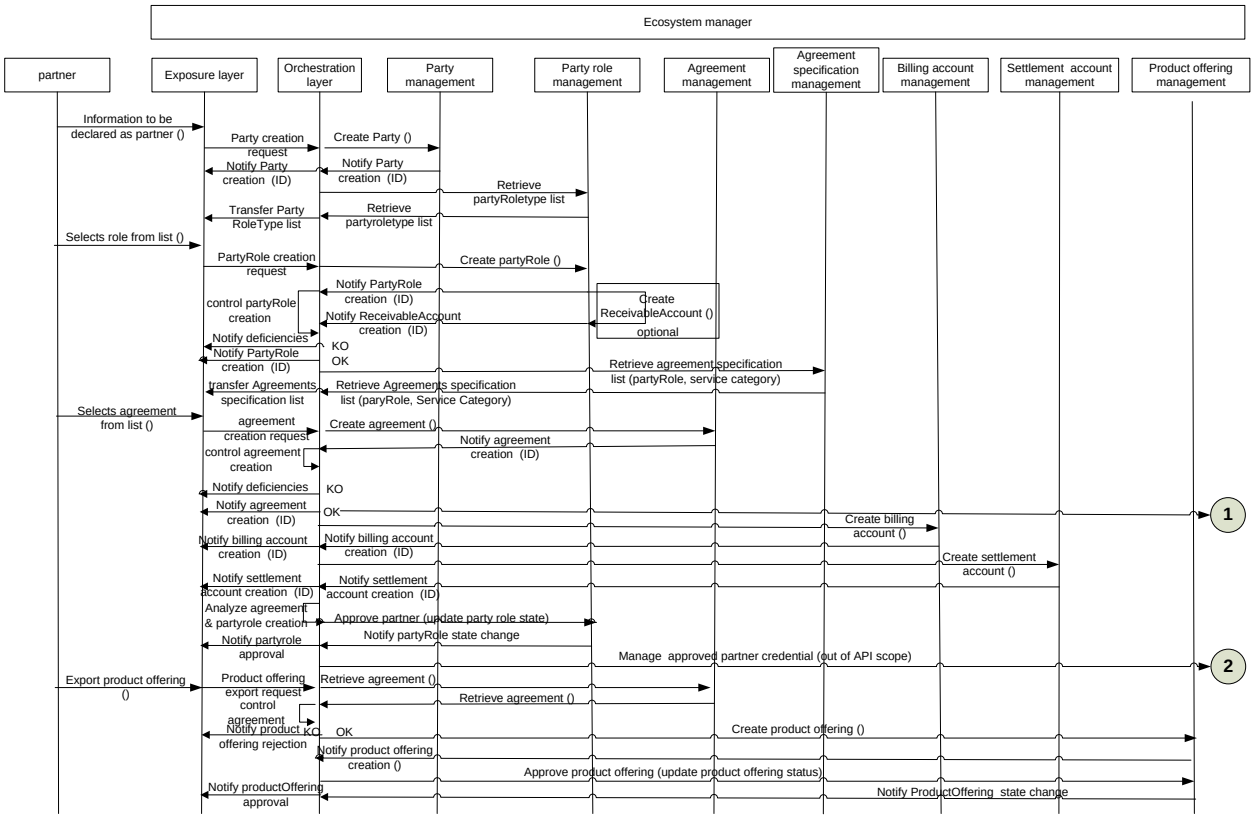
- Prospective partner is not known by ecosystem manager system
- Party resources is created
- List of partyRoleType is retrieved
- Prospective partner selects a partyRoleType
- PartyRole is created and receivable account can be optionally created
- Business logic controls party role creation and notifies exposure layer of partyRole creation or deficiencies if there is errors or inconsistencies
- Agreement, billing account and / or settlement account resources are created
- Agreement creation can be performed several times
- Prospective partner validate agreement and, then, agreement, billing account and / or settlement account are updated
- Business logic controls agreement creation and notifies exposure layer of agreement creation or deficiencies if there is errors or inconsistencies
- Partner is approved if its approval depends on partyRole and agreement creation
- Business logic manage partner credentials (out of API scope)
- Business logic checks if partner productOffering creation request is covered by an agreement
- Partner creates a product offering if allowed
- Ecosystem manager approves or rejects product offering creation.

Note:

- For the sake of readability, Partner subsequent action in case of deficiencies when controlling partyRole or Agreement creation are not presented on the sequence diagram.

Option 3: party role / agreement resources creation follows interaction with prospective partner

(Agreement is selected by prospective partner)



Partner On Boarding
Option 3: Party role / agreement resources creation follows interaction with prospective partner
(agreement is selected by prospective partner)

This sequence diagram describes a partner on boarding process where:

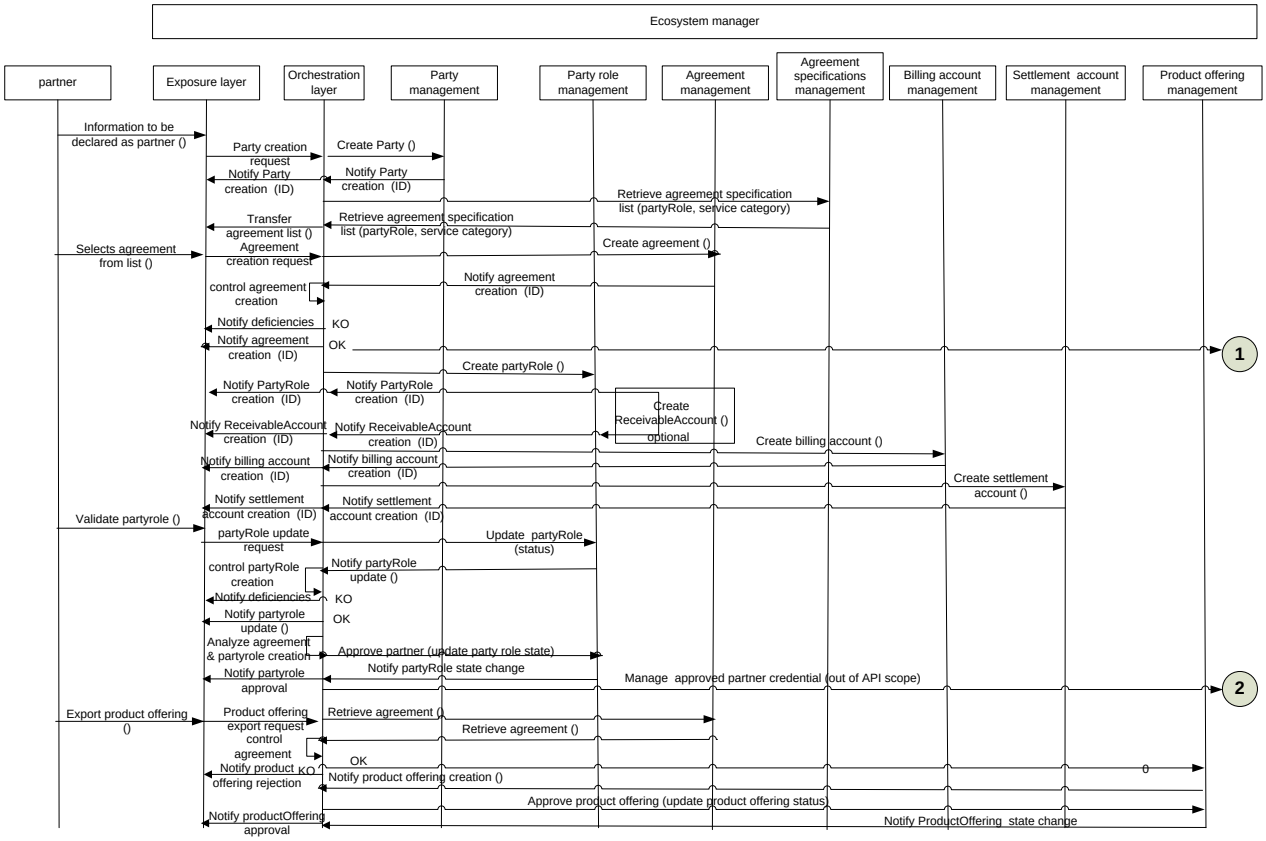
- Prospective partner is not known by ecosystem manager system
- Party resources is created
- List of partyRoleType is retrieved
- Prospective partner selects a partyRoleType
- PartyRole is created and receivable account can be optionally created
- Business logic controls party role creation and notifies exposure layer of partyRole creation or deficiencies if there is errors or inconsistencies
- List of agreement specifications is retrieved based on partyRoleType and/or service category
- Prospective partner selects an agreement specification
- Agreement is created
- Business logic controls agreement creation and notifies exposure layer of agreement creation or deficiencies if there is errors or inconsistencies
- Agreement creation can be performed several times
- Billing account and / or settlement account resources are created
- Partner is approved if its approval depends on partyRole and agreement creation
- Business logic manage partner credentials (out of API scope)
- Business logic checks if partner productOffering creation request is covered by an agreement
- Partner creates a product offering if allowed
- Ecosystem manager approves or rejects product offering creation.

Note:

- For the sake of readability, Partner subsequent action in case of deficiencies when controlling partyRole or Agreement creation are not presented on the sequence diagram.

Option 4: agreement / party role resources creation follows interaction with prospective partner

(Party role is selected automatically based upon agreement chosen)



Partner On Boarding
Option 4 : agreement / Party role resources creation follows interaction with prospective partner
(partyrole is selected automatically based upon agreement chosen)

This sequence diagram describes a partner on boarding process where:

- Prospective partner is not known by ecosystem manager system
- Party resources is created
- List of agreement specifications is retrieved based on partyRoleType and/or service category
- Prospective partner selects an agreement specification
- Agreement is created
- Agreement creation can be performed several times
- Business logic controls agreement creation and notifies exposure layer of agreement creation or deficiencies if there is errors or inconsistencies
- Billing account and / or settlement account resources are created
- PartyRole is created depending on agreement specification selected and receivableAccount is optionally created
- prospective partner validates partyRole and PartyRole is updated

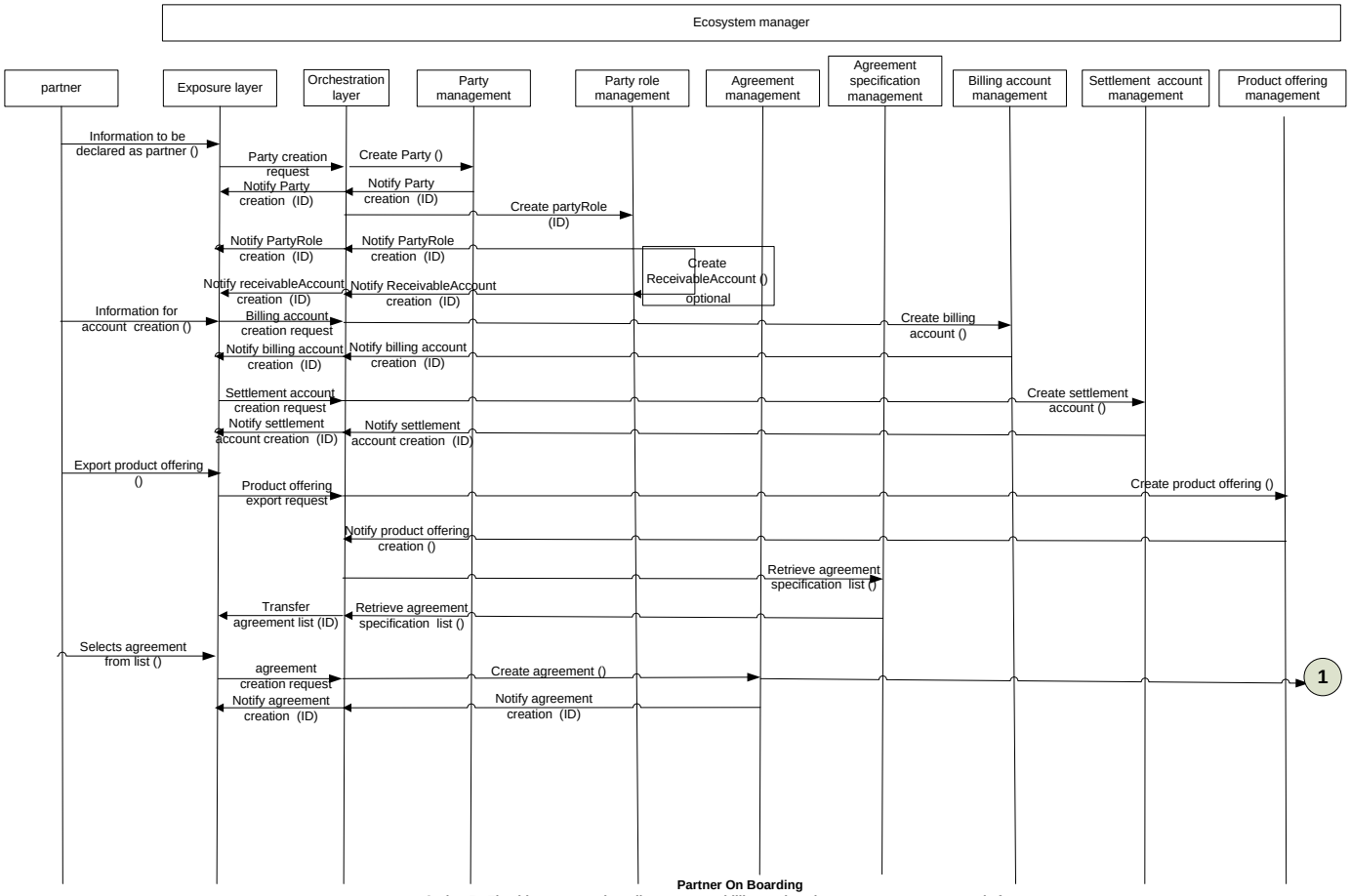
- Business logic controls party role creation and notifies exposure layer of partyRole creation or deficiencies if there is errors or inconsistencies
- Partner is approved if its approval depends on partyRole and agreement creation
- Business logic manage partner credentials (out of API scope)
- Business logic checks if partner productOffering creation request is covered by an agreement
- Partner creates a product offering if allowed
- Ecosystem manager approves or rejects product offering creation.

Note:

- For the sake of readability, Partner subsequent action in case of deficiencies when controlling partyRole or Agreement creation are not presented on the sequence diagram.

Option 5: Ehealth partner on boarding use case: billing and settlement accounts are created after party

(Then service information is provided and agreement is created)



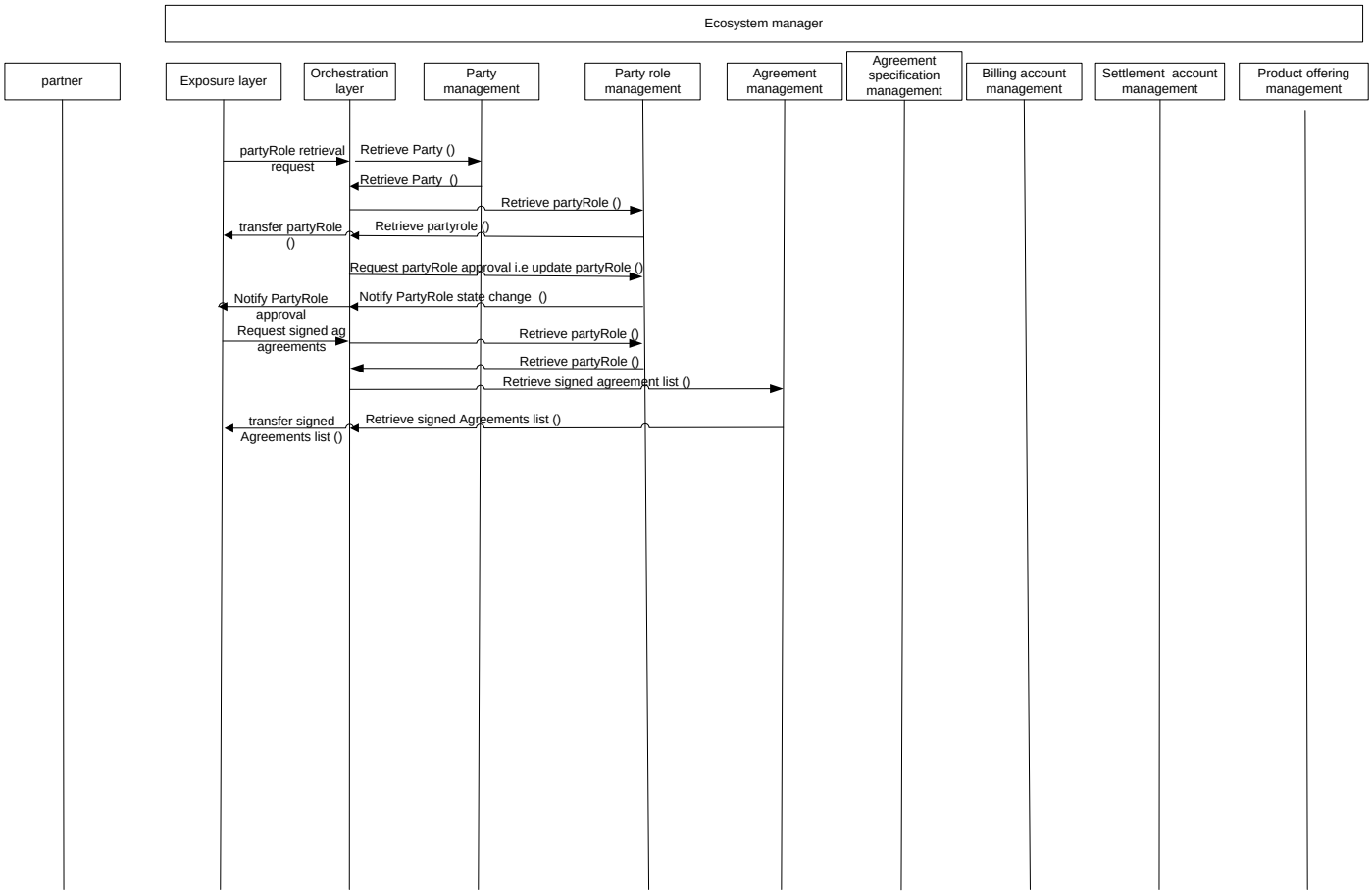
Partner On Boarding
Option 5 : Ehealth partner on boarding use case: billing and settlement accounts are created after party.
(Then service information is provided and agreement is created)

This sequence diagram describes the eHealth partner on boarding process where:

- prospective partner is not known by ecosystem manager system
- Creation of partner includes creation of Party, PartyRole and, optionally, ReceivableAccount
- Creation of billing account is triggered by an account creation request from partner
- “Service info” provision by partner triggers product offering creation

- Creation of product offering triggers agreement list notification.

Option 6: party has already selected a partyRole, partyRole needs to be approved, Agreement signed by partyRole needs to be retrieved

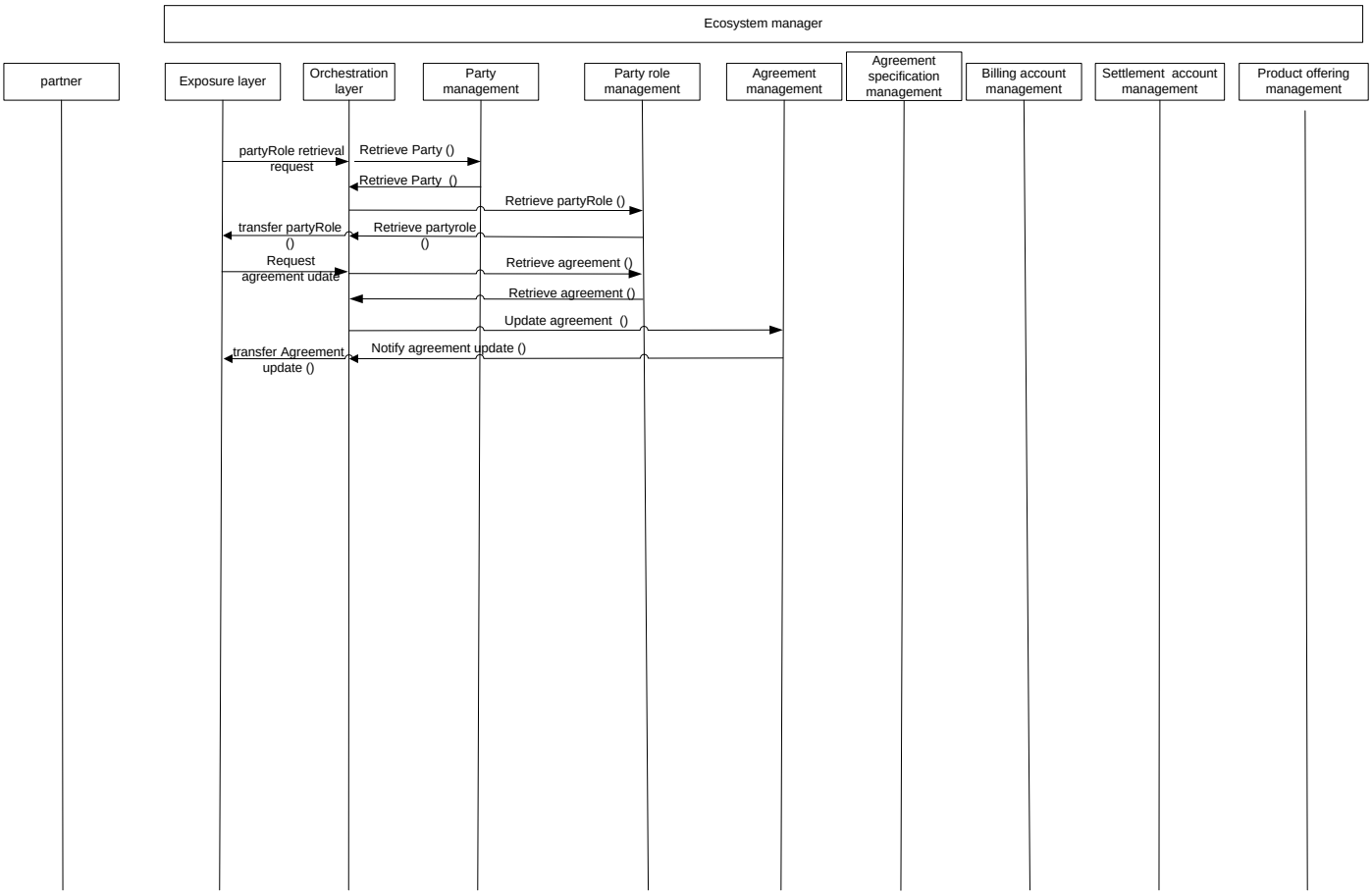


Partner On Boarding
Option 6: party has already selected a partyRole, partyRole needs to be approved, Agreement signed by partyRole needs to be retrieved

This sequence diagram describes a partner on boarding process where:

- prospective partner already exists and needs to be approved
- If partyRole identifier is not known
  - Party is retrieved (relevance to be confirmed)
  - partyRole is retrieved
- If partyrole identifier is known
  - partyRole is retrieved
- Ecosystem manager request partyRole approval
  - partyRole status is updated
  - partyRole state change is notified
- list of agreements signed by partner (partyRole) are retrieved

Option 7: partyRole exist, Agreement is updated

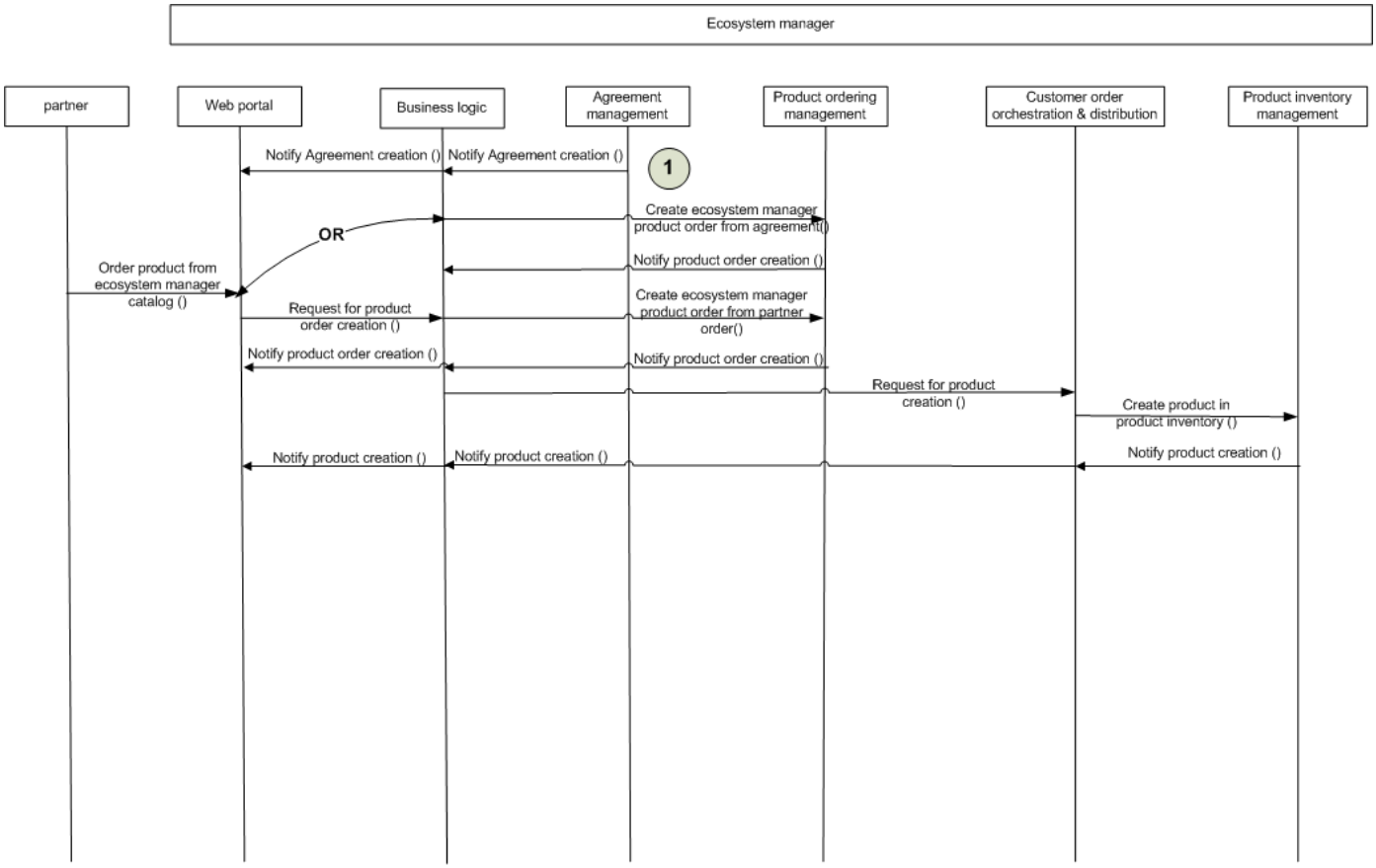


Partner On Boarding
Option 7: partyRole exist, Agreement is updated

This sequence diagram describes a partner on boarding process where:

- prospective partner already exists and agreement needs to be updated
- If partyRole identifier is not known
  - Party is retrieved (relevance to be confirmed)
  - partyRole is retrieved
- If partyrole identifier is known
  - partyRole is retrieved
- list of agreements related to party role are retrieved
- agreement is updated.

Ordering of an ecosystem manager product by partner



Partner On Boarding
Ordering of an ecosystem manager product by partner

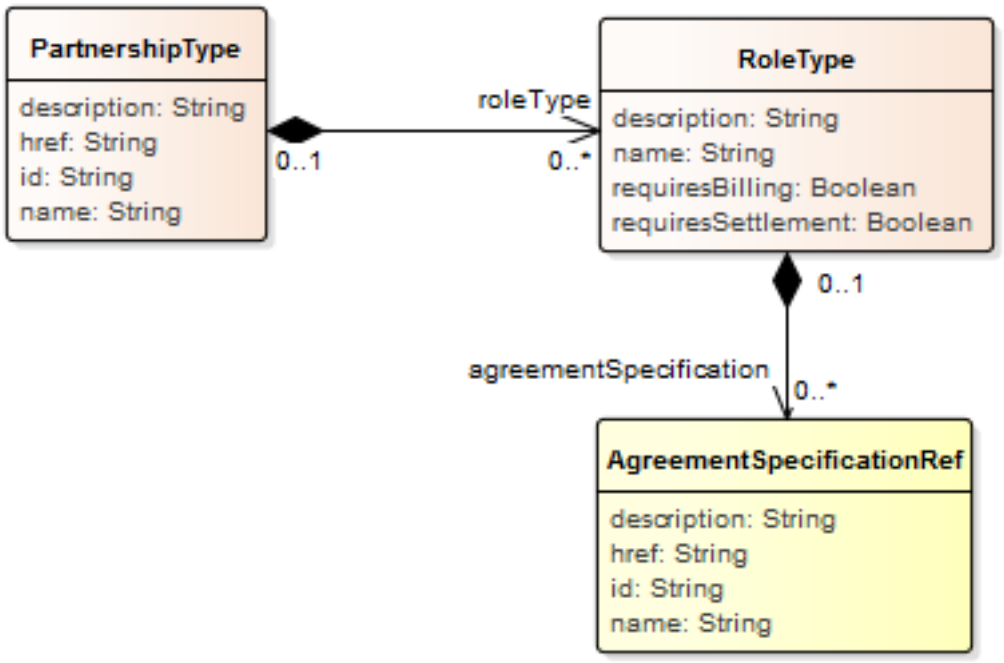
RESOURCE MODEL

Managed Entity and Task Resource Models

PARTNERSHIP TYPE RESOURCE

A partnership type contains all the information for the setup of a partnership of a given kind. This includes the list of identified role types for the partnership with the corresponding agreement specifications.

Resource model



Field descriptions

PartnershipType fields

description	A string. An explanatory text regarding this partnership type.
href	A string. The reference url for this partnership type.
id	A string. The identifier of the partnership type.
name	A string. An identifying name for the partnership type.
roleType	A list of role types (RoleType [*]). A RoleType represents the type of a PartyRole, defined in the context of a given type of partnership, such as Buyer, Seller.

RoleType sub-resource

A RoleType represents the type of a PartyRole, defined in the context of a given type of partnership, such as Buyer, Seller.

description	A string. An explanatory text documenting the role type.
name	A string. The name of the role type.
requiresBilling	A boolean. Indicates whether billing operations will be associated to parties playing the role.
requiresSettlement	A boolean. Indicates whether settlement operations will be associated to parties playing the role.
agreementSpecification	A list of agreement specification references (AgreementSpecificationRef [*]). An AgreementSpecification represents a template of an agreement that can be used when establishing partnerships.

Json representation sample

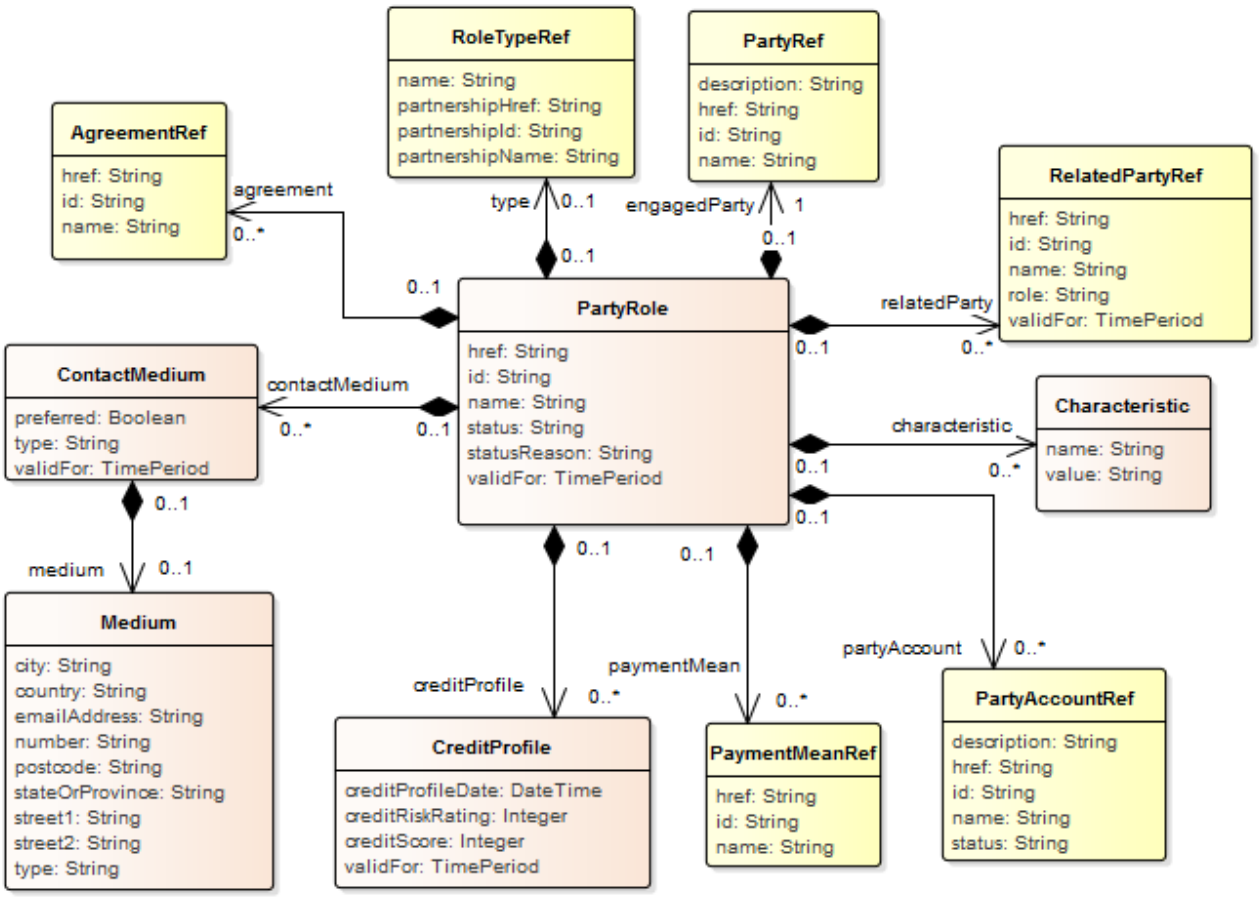
We provide below the json representation of an example of a 'PartnershipType' resource object


```
{
  "description": "This  partnership type ...",
  "href": "https://host:port/onboardingManagement/partnershipType/6837",
  "id": "6837",
  "name": "Dream Partnership",
  "roleType": [
    [
      {
        "name": "ContentProvider",
        "agreementSpecification": [
          {
            "name": "ContentLicenseAgreement",
            "id": "33"
          }
        ]
      },
      {
        "name": "CloudProvider"
      },
      {
        "name": "Developer",
        "agreementSpecification": [
          {
            "name": "ProfitShareAgreement",
            "id": "32"
          }
        ]
      },
      {
        "name": "Tester"
      }
    ]
  ]
}
```

PARTY ROLE RESOURCE

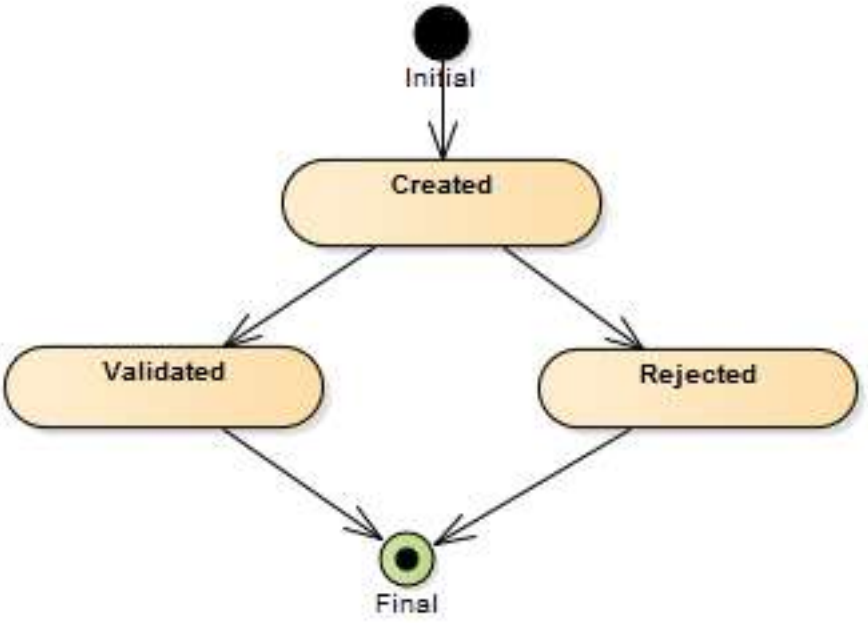
The part played by a party in a given context.

Resource model



Lifecycle

The lifecycle of a party role is tracked by its status field. Typical values are Created, Validated and Rejected. Validated status occurs when a potential/prospective partner accepts to follow the party role suggested to him, whereas Rejected status occurs when the prospective partner rejects the assigned party role. The state machine specifying the typical state change transitions is provided below.



Field descriptions

PartyRole fields

href	A string. Url used to reference the party role.
id	A string. Unique identifier for PartyRoles.
name	A string. A word, term, or phrase by which the PartyRole is known and distinguished from other PartyRoles.
status	A string. Used to track the lifecycle status of the party role.
statusReason	A string. A string providing an explanation on the value of the status lifecycle. For instance if the status is Rejected, statusReason will provide the reason for rejection.
validFor	A time period. The time period that the PartyRole is valid for.
engagedParty	A party reference (PartyRef). A party represents an organization or an individual.
type	A role type reference (RoleTypeRef).
partyAccount	A list of party account references (PartyAccountRef [*]). A party account is an arrangement that a party has with an enterprise that provides products to the party.
paymentMean	A list of payment mean references (PaymentMeanRef [*]). A payment mean defines a specific mean of payment (e.g direct debit with all details associated).

contactMedium	A list of contact mediums (ContactMedium [*]). Indicates the contact medium that could be used to contact the party.
characteristic	A list of characteristics (Characteristic [*]). Describes a given characteristic of an object or entity through a name/value pair.
creditProfile	A list of credit profiles (CreditProfile [*]). Credit profile for the party (containing credit scoring, ...). By default only the current credit profile is retrieved. It can be used as a list to give the party credit profiles history, the first one in the list will be the current one.
agreement	A list of agreement references (AgreementRef [*]). An agreement represents a contract or arrangement, either written or verbal and sometimes enforceable by law, such as a service level agreement or a customer price agreement. An agreement involves a number of other business entities, such as products, services, and resources and/or their specifications.
relatedParty	A list of related party references (RelatedPartyRef [*]). A related party defines party or party role linked to a specific entity.

ContactMedium sub-resource

Indicates the contact medium that could be used to contact the party.

preferred	A boolean. If true, indicates that is the preferred contact medium.
type	A string. Type of the contact medium, such as: email address, telephone number, postal address.
validFor	A time period. The time period that the contact medium is valid for.
medium	A medium (Medium). Describes the contact medium that could be used to contact a party (an individual or an organization).

Characteristic sub-resource

Describes a given characteristic of an object or entity through a name/value pair.

name	A string. Name of the characteristic.
value	A string. Value of the characteristic.

CreditProfile sub-resource

Credit profile for the party (containing credit scoring, ...). By default only the current credit profile is retrieved. It can be used as a list to give the party credit profiles history, the first one in the list will be the current one.

creditProfileDate	A date time (DateTime). The date the profile was established.
creditRiskRating	An integer. This is an integer whose value is used to rate the risk.
creditScore	An integer. A measure of a person's or an organization's creditworthiness

	calculated on the basis of a combination of factors such as their income and credit history.
validFor	A time period. The period for which the profile is valid.

PartyRef relationship

Party reference. A party represents an organization or an individual.

description	A string. Text describing the referred party.
href	A string. Reference of the referred party (such as a partner or any other party role).
id	A string. Unique identifier of the referred party.
name	A string. Name of the referred party (such as a partner or any other party role).

RoleTypeRef relationship

RoleType reference.

name	A string. The name of the role type. It uniquely identifies the role type within the partnership type.
partnershipHref	A string. Reference url of the partnership type containing the role type.
partnershipId	A string. The identifier of the partnership type containing the role type.
partnershipName	A string. The name of the partnership type defining this role type.

PartyAccountRef relationship

PartyAccount reference. A party account is an arrangement that a party has with an enterprise that provides products to the party.

description	A string. Detailed description of the party account.
href	A string. Reference of the party account.
id	A string. Unique identifier of the party account.
name	A string. Name of the party account.
status	A string. The condition of the account, such as due, paid, in arrears.

PaymentMeanRef relationship

PaymentMean reference. A payment mean defines a specific mean of payment (e.g direct debit with all details associated).

href	A string. Reference of the payment mean.
id	A string. Unique identifier of the payment mean.

name A string. Name of the payment mean.

AgreementRef relationship

Agreement reference. An agreement represents a contract or arrangement, either written or verbal and sometimes enforceable by law, such as a service level agreement or a customer price agreement. An agreement involves a number of other business entities, such as products, services, and resources and/or their specifications.

href A string. Reference of the agreement.

id A string. Identifier of the agreement.

name A string. Name of the agreement.

RelatedPartyRef relationship

RelatedParty reference. A related party defines party or party role linked to a specific entity.

href A string. Reference of the related party, could be a party reference or a party role reference.

id A string. Unique identifier of a related party.

name A string. Name of the related party.

role A string. Role of the related party.

validFor A time period. Validity period of the related party.

Json representation sample

We provide below the json representation of an example of a 'PartyRole' resource object


```
{
  "href": "https://host:port/onboardingManagement/partyRole/5553",
  "id": "5553",
  "name": "Supplier",
  "status": "Created",
  "statusReason": "",
  "validFor": {
    "startDateTime": "2016-04-04T00:00",
    "endDateTime": "2016-11-03T00:00"
  },
  "engagedParty": {
    "description": "This  party ...",
    "href": "https://host:port/onboardingManagement/party/5927",
    "id": "5927",
    "name": "Booking Corporation"
  },
  "type": {
    "partnershipId": "178",
    "name": "Supplier",
    "partnershipName": "Application provider partnership"
  },
  "partyAccount": [
```



```
    {
      "description": "This  party account ...",
      "href": "https://host:port/onboardingManagement/partyAccount/9232",
      "id": "9232",
      "name": "Travel account",
      "status": "In Arrears"
    }
  ],
  "paymentMean": [
    {
      "href": "https://host:port/onboardingManagement/paymentMean/2008",
      "id": "2008",
      "name": "family payment"
    }
  ],
  "contactMedium": [
    {
      "preferred": true,
      "type": "PostalAddress",
      "validFor": {
        "startDateTime": "2016-04-11T00:00",
        "endDateTime": "2016-11-03T00:00"
      },
      "medium": {
        "city": "Paris",
        "country": "France",
        "emailAddress": "coco.chanel@orange.fr",
        "number": "+336641234567",
        "postcode": "75000",
        "stateOrProvince": "Ile de France",
        "street1": "Rue Picasso",
        "street2": "",
        "type": "home"
      }
    }
  ],
  "characteristic": [
    [
      {
        "name": "Colour",
        "value": "pink"
      },
      {
        "name": "Memory",
        "value": "64"
      }
    ]
  ],
  "creditProfile": [
    {
      "creditProfileDate": "2016-04-08T00:00",
      "creditRiskRating": 1,
      "creditScore": 5,
      "validFor": {
        "startDateTime": "2016-04-06T00:00",
        "endDateTime": "2016-11-03T00:00"
```

```
        }
      }
    ],
    "agreement": [
      {
        "href": "https://host:port/onboardingManagement/agreement/8812",
        "id": "8812",
        "name": "Summer Contract Agreement"
      }
    ],
    "relatedParty": [
      {
        "href": "https://host:port/onboardingManagement/relatedParty/3532",
        "id": "3532",
        "name": "Gustave Flaubert",
        "role": "seller",
        "validFor": {
          "startDateTime": "2016-04-06T00:00",
          "endDateTime": "2016-11-03T00:00"
        }
      }
    ]
}
```

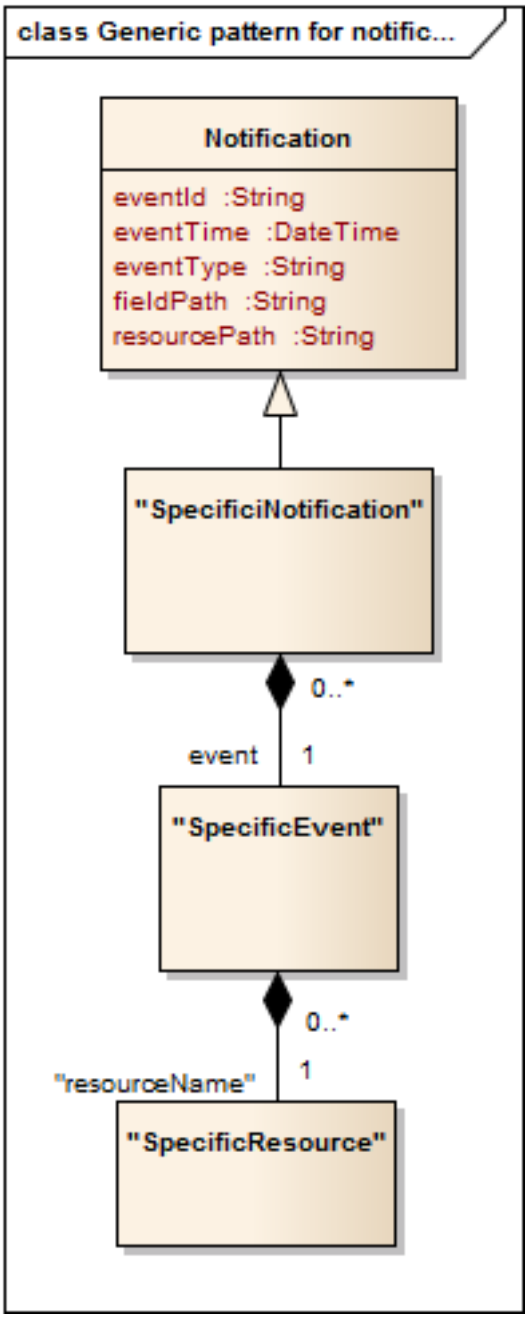

Notification Resource Models

6 notifications are defined for this API

Notifications related to PartnershipType:
- PartnershipTypeCreationNotification
- PartnershipTypeRemoveNotification

Notifications related to PartyRole:
- PartyRoleCreationNotification
- PartyRoleAttributeValueChangeNotification
- PartyRoleStateChangeNotification
- PartyRoleRemoveNotification

The notification structure for all notifications in this API follow the pattern depicted by the figure below.
A notification resource (depicted by "SpecificNotification" placeholder) is a sub class of a generic Notification structure containing an id of the event occurence (eventId), an event timestamp (eventTime), and the name of the notification resource (eventType).
This notification structure owns an event structure ("SpecificEvent" placeholder) linked to the resource concerned by the notification using the resource name as access field ("resourceName" placeholder).



PARTNERSHIP TYPE CREATION NOTIFICATION

Notification sent when a new PartnershipType resource is created.

Json representation sample

We provide below the json representation of an example of a 'PartnershipTypeCreationNotification' notification object

```
{
   "eventId":"00001",
   "eventTime":"2015-11-16T16:42:25-04:00",
   "eventType":"PartnershipTypeCreationNotification",
    "event": {
```

```
    "partnershipType" :
      {-- SEE PartnershipType RESOURCE SAMPLE --}
  }
}
```

PARTNERSHIP TYPE REMOVE NOTIFICATION

Notification sent when removing a PartnershipType resource.

Json representation sample

We provide below the json representation of an example of a 'PartnershipTypeRemoveNotification' notification object

```
{
  "eventId":"00001",
  "eventTime":"2015-11-16T16:42:25-04:00",
  "eventType":"PartnershipTypeRemoveNotification",
   "event": {
     "partnershipType" :
       {-- SEE PartnershipType RESOURCE SAMPLE --}
  }
}
```

PARTY ROLE CREATION NOTIFICATION

Notification sent when a new PartyRole resource is created.

Json representation sample

We provide below the json representation of an example of a 'PartyRoleCreationNotification' notification object

```
{
  "eventId":"00001",
  "eventTime":"2015-11-16T16:42:25-04:00",
  "eventType":"PartyRoleCreationNotification",
   "event": {
     "partyRole" :
       {-- SEE PartyRole RESOURCE SAMPLE --}
  }
}
```

PARTY ROLE ATTRIBUTE VALUE CHANGE NOTIFICATION

Notification sent when changing an attribute of a PartyRole resource.

Json representation sample

We provide below the json representation of an example of a 'PartyRoleAttributeValueChangeNotification' notification object

```
{
  "eventId":"00001",
  "eventTime":"2015-11-16T16:42:25-04:00",
  "eventType":"PartyRoleAttributeValueChangeNotification",
   "event": {
     "partyRole" :
       {-- SEE PartyRole RESOURCE SAMPLE --}
  }
}
```

PARTY ROLE STATE CHANGE NOTIFICATION

Notification sent when changing the state of a PartyRole resource.

Json representation sample

We provide below the json representation of an example of a 'PartyRoleStateChangeNotification' notification object

```
{
  "eventId":"00001",
  "eventTime":"2015-11-16T16:42:25-04:00",
  "eventType":"PartyRoleStateChangeNotification",
   "event": {
     "partyRole" :
       {-- SEE PartyRole RESOURCE SAMPLE --}
  }
}
```

PARTY ROLE REMOVE NOTIFICATION

Notification sent when removing a PartyRole resource.

Json representation sample

We provide below the json representation of an example of a 'PartyRoleRemoveNotification' notification object

```
{
  "eventId":"00001",
  "eventTime":"2015-11-16T16:42:25-04:00",
  "eventType":"PartyRoleRemoveNotification",
   "event": {
     "partyRole" :
       {-- SEE PartyRole RESOURCE SAMPLE --}
  }
}
```

API OPERATIONS

Remember the following Uniform Contract:

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource.
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute Task Resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods.

Filtering and attribute selection rules are described in the TMF REST Design Guidelines.

Notifications are also described in a subsequent section.

OPERATIONS ON PARTNERSHIP TYPE

LIST PARTNERSHIP TYPES

```
GET /partnershipType?fields=...&{filtering}
```

Description

This operation list partnership type entities.
Attribute selection is enabled for all first level attributes.
Filtering may be available depending on the compliance level supported by an implementation.

Usage Samples

Here's an example of a request for retrieving PartnershipType resources.

Request

```
GET /onboardingManagement/partnershipType
Accept: application/json
```

Response

```
200

[
{
   "description": "This  partnership type ...",
   "href": "https://host:port/onboardingManagement/partnershipType/6837",
   "id": "6837",
   "name": "Dream Partnership",
   "roleType": [
      [
         {
            "name": "ContentProvider",
            "agreementSpecification": [
               {
                  "name": "ContentLicenseAgreement",
                  "id": "33"
               }
            ]
         },
         {
            "name": "CloudProvider"
         },
         {
            "name": "Developer",
            "agreementSpecification": [
               {
                  "name": "ProfitShareAgreement",
                  "id": "32"
               }
            ]
         },
         {
            "name": "Tester"
         }
```

```
    ]
  ]
}
]
```

RETRIEVE PARTNERSHIP TYPE

`GET /partnershipType/{id}?fields=...&{filtering}`

Description

This operation retrieves a partnership type entity.
Attribute selection is enabled for all first level attributes.
Filtering on sub-resources may be available depending on the compliance level supported by an implementation.

Usage Samples

Here's an example of a request for retrieving a PartnershipType resource.

Request

```
GET /onboardingManagement/partnershipType/6837
Accept: application/json
```

Response

```
200

{
   "description": "This  partnership type ...",
   "href": "https://host:port/onboardingManagement/partnershipType/6837",
   "id": "6837",
   "name": "Dream Partnership",
   "roleType": [
      [
         {
            "name": "ContentProvider",
            "agreementSpecification": [
               {
                  "name": "ContentLicenseAgreement",
                  "id": "33"
               }
            ]
         },
         {
            "name": "CloudProvider"
         },
```

```
        {
          "name": "Developer",
          "agreementSpecification": [
            {
              "name": "ProfitShareAgreement",
              "id": "32"
            }
          ]
        },
        {
          "name": "Tester"
        }
      ]
    ]
}
```


CREATE PARTNERSHIP TYPE

POST /partnershipType

Note: this operation is available only to ADMIN API users

Description

This operation creates a partnership type entity.

Mandatory and Non Mandatory Attributes

The following tables provides the list of mandatory and non mandatory attributes when creating a PartnershipType, including any possible rule conditions and applicable default values.

Mandatory Attributes	Rule
name	

Non Mandatory Attributes	Default Value	Rule
description		
roleType		

Additional Rules

The following table provides additional rules indicating mandatory fields in sub-resources or relationships when creating a PartnershipType resource.

Context	Mandatory Sub-Attributes
roleType	name

Usage Samples

Here's an example of a request for creating a PartnershipType resource. In this example the request only passes mandatory attributes.

Request

POST /onboardingManagement/partnershipType
Content-Type: application/json
 	
{
"name": "Dream Partnership"
}

Response

201
 	
{
"href": "https://host:port/onboardingManagement/partnershipType/6837",
"id": "6837",
"name": "Dream Partnership"
}

PATCH PARTNERSHIP TYPE

`PATCH /partnershipType/{id}`

Note: this operation is available only to ADMIN API users

Description

This operation allows partial updates of a partnership type entity. Support of json/merge (https://tools.ietf.org/html/rfc7386) is mandatory, support of json/patch (http://tools.ietf.org/html/rfc5789) is optional.

Note: If the update operation yields to the creation of sub-resources or relationships, the same rules concerning mandatory sub-resource attributes and default value settings in the POST operation applies to the PATCH operation. Hence these tables are not repeated here.

Patchable and Non Patchable Attributes

The tables below provide the list of patchable and non patchable attributes, including constraint rules on their usage.

Patchable Attributes	Rule
name	
description	
roleType	

Non Patchable Attributes	Rule
href	
id	

Usage Samples

Here's an example of a request for patching a PartnershipType resource.

Request

```
PATCH /onboardingManagement/partnershipType/6837
Content-Type: application/merge-patch+json

{
   "name": "new name"
}
```

Response

```
201

{
   "description": "This  partnership type ...",
   "href": "https://host:port/onboardingManagement/partnershipType/6837",
   "id": "6837",
   "name": "new name",
   "roleType": [
      [
         {
            "name": "ContentProvider",
            "agreementSpecification": [
               {
                  "name": "ContentLicenseAgreement",
                  "id": "33"
               }
            ]
         },
         {
            "name": "CloudProvider"
         },
         {
            "name": "Developer",
            "agreementSpecification": [
               {
                  "name": "ProfitShareAgreement",
                  "id": "32"
               }
            ]
         },
         {
            "name": "Tester"
         }
      ]
   ]
}
```

DELETE PARTNERSHIP TYPE

`DELETE /partnershipType/{id}`

Note: this operation is available only to ADMIN API users

Description

This operation deletes a partnership type entity.

Usage Samples

Here's an example of a request for deleting a PartnershipType resource.

Request
DELETE /onboardingManagement/partnershipType/42
Response
204

OPERATIONS ON PARTY ROLE

LIST PARTY ROLES

`GET /partyRole?fields=...&{filtering}`

Description

This operation list party role entities.
Attribute selection is enabled for all first level attributes.
Filtering may be available depending on the compliance level supported by an implementation.

Usage Samples

Here's an example of a request for retrieving PartyRole resources.

Retrieving all party roles linked to a given engaged party (named 'GrooveDotCom'). The result items are shrinked to show only the id and the name (fields=id,name)

Request

```
GET /onboardingManagement/partyRole?fields=id,name,engagedParty.name&engagedParty.name="GrooveDotCom"
Accept: application/json
```

Response

```
200

[
   {
      "engagedParty": {
         "name": "GrooveDotCom"
      },
      "id": "6756",
      "name": "Music Seller"
   },
   {
      "engagedParty": {
         "name": "GrooveDotCom"
      },
      "id": "4231",
      "name": "Software Provider"
   }
]
```

RETRIEVE PARTY ROLE

`GET /partyRole/{id}?fields=...&{filtering}`

Description

This operation retrieves a party role entity.
Attribute selection is enabled for all first level attributes.
Filtering on sub-resources may be available depending on the compliance level supported by an implementation.

Usage Samples

Here's an example of a request for retrieving a PartyRole resource.

Request

```
GET /onboardingManagement/partyRole/5553
Accept: application/json
```

Response

200


```
{
   "href": "https://host:port/onboardingManagement/partyRole/5553",
   "id": "5553",
   "name": "Supplier",
   "status": "Created",
   "statusReason": "",
   "validFor": {
      "startDateTime": "2016-04-04T00:00",
      "endDateTime": "2016-11-03T00:00"
   },
   "engagedParty": {
      "description": "This  party ...",
      "href": "https://host:port/onboardingManagement/party/5927",
      "id": "5927",
      "name": "Booking Corporation"
   },
   "type": {
      "partnershipId": "178",
      "name": "Supplier",
      "partnershipName": "Application provider partnership"
   },
   "partyAccount": [
      {
         "description": "This  party account ...",
         "href": "https://host:port/onboardingManagement/partyAccount/9232",
         "id": "9232",
         "name": "Travel account",
         "status": "In Arrears"
      }
   ],
   "paymentMean": [
      {
         "href": "https://host:port/onboardingManagement/paymentMean/2008",
         "id": "2008",
         "name": "family payment"
      }
   ],
   "contactMedium": [
      {
         "preferred": true,
         "type": "PostalAddress",
         "validFor": {
            "startDateTime": "2016-04-11T00:00",
            "endDateTime": "2016-11-03T00:00"
         },
         "medium": {
            "city": "Paris",
            "country": "France",
            "emailAddress": "coco.chanel@orange.fr",
            "number": "+336641234567",
            "postcode": "75000",
```

```
            "stateOrProvince": "Ile de France",
            "street1": "Rue Picasso",
            "street2": "",
            "type": "home"
          }
        }
    ],
    "characteristic": [
      [
        {
          "name": "Colour",
          "value": "pink"
        },
        {
          "name": "Memory",
          "value": "64"
        }
      ]
    ],
    "creditProfile": [
      {
        "creditProfileDate": "2016-04-08T00:00",
        "creditRiskRating": 1,
        "creditScore": 5,
        "validFor": {
          "startDateTime": "2016-04-06T00:00",
          "endDateTime": "2016-11-03T00:00"
        }
      }
    ],
    "agreement": [
      {
        "href": "https://host:port/onboardingManagement/agreement/8812",
        "id": "8812",
        "name": "Summer Contract Agreement"
      }
    ],
    "relatedParty": [
      {
        "href": "https://host:port/onboardingManagement/relatedParty/3532",
        "id": "3532",
        "name": "Gustave Flaubert",
        "role": "seller",
        "validFor": {
          "startDateTime": "2016-04-06T00:00",
          "endDateTime": "2016-11-03T00:00"
        }
      }
    ]
}
```


CREATE PARTY ROLE

POST /partyRole

Note: this operation is available only to ADMIN API users

Description

This operation creates a party role entity.

Mandatory and Non Mandatory Attributes

The following tables provides the list of mandatory and non mandatory attributes when creating a PartyRole, including any possible rule conditions and applicable default values.

Mandatory Attributes	Rule
name	
type	

Non Mandatory Attributes	Default Value	Rule
status		
statusReason		
validFor		
engagedParty	Automatically generated	
partyAccount		
paymentMean		
contactMedium		
characteristic		
creditProfile		
agreement		
relatedParty		

Additional Rules

The following table provides additional rules indicating mandatory fields in sub-resources or relationships when creating a PartyRole resource.

Context	Mandatory Sub-Attributes
engagedParty	id
characteristic	name, value
contactMedium	type, medium
partyAccount	id, name, status
creditProfile	creditProfileDate, validFor
paymentMean	id, href

Default Values Summary

When creating the resource, the following table summarizes the default values applicable to optional attributes of the resource (or sub-resources).

Attributes	Default Value
id	Automatically generated
engagedParty	Automatically generated

Usage Samples

Here's an example of a request for creating a PartyRole resource. In this example the request only passes mandatory attributes.

Request

```
POST /onboardingManagement/partyRole
Content-Type: application/json

{
   "name": "Supplier",
   "type": {
      "partnershipId": "178",
      "name": "Supplier",
      "partnershipName": "Application provider partnership"
   }
}
```

Response

```
201

{
   "href": "https://host:port/onboardingManagement/partyRole/5553",
   "id": "5553",
   "name": "Supplier",
   "engagedParty": {
      "description": "This  party ...",
      "href": "https://host:port/onboardingManagement/party/5927",
      "id": "5927",
      "name": "Booking Corporation"
   },
   "type": {
      "partnershipId": "178",
      "name": "Supplier",
      "partnershipName": "Application provider partnership"
   }
}
```

PATCH PARTY ROLE

`PATCH /partyRole/{id}`

Description

This operation allows partial updates of a party role entity. Support of json/merge (https://tools.ietf.org/html/rfc7386) is mandatory, support of json/patch (http://tools.ietf.org/html/rfc5789) is optional.

Note: If the update operation yields to the creation of sub-resources or relationships, the same rules concerning mandatory sub-resource attributes and default value settings in the POST operation applies to the PATCH operation. Hence these tables are not repeated here.

Patchable and Non Patchable Attributes

The tables below provide the list of patchable and non patchable attributes, including constraint rules on their usage.

Patchable Attributes	Rule
name	
status	
statusReason	
validFor	
engagedParty	
type	
partyAccount	
paymentMean	
contactMedium	
characteristic	
creditProfile	
agreement	
relatedParty	

Non Patchable Attributes	Rule
id	
href	
name	
status	
statusReason	
validFor	
engagedParty	
type	
partyAccount	
paymentMean	
contactMedium	
characteristic	
creditProfile	
agreement	
relatedParty	

Usage Samples

Here's an example of requests for patching a PartyRole resource.

Changing the status to 'prospective' (using json-merge)

Request
PATCH /onboardingManagement/partyRole/42

Content-Type: application/merge-patch+json
 	
{
"status": "prospective"
}

Response
201 { Similar JSON response as in GET response with status added or changed }

Changing the status to 'prospective' (using json-patch)

Request

PATCH /onboardingManagement/partyRole/42
Content-Type: application/json-patch+json
 	
{
"path": "/status",
"value": "prospective",
"op": "replace"
}

Response
201 { Similar JSON response as in GET response with status added or changed }

DELETE PARTY ROLE

`DELETE /partyRole/{id}`

Note: this operation is available only to ADMIN API users

Description

This operation deletes a party role entity.

Usage Samples

Here's an example of a request for deleting a PartyRole resource.

Request
DELETE /onboardingManagement/partyRole/42
Response
204

API NOTIFICATIONS

For every single of operation on the entities use the following templates and provide sample REST notification POST calls.

It is assumed that the Pub/Sub uses the Register and UnRegister mechanisms described in the REST Guidelines reproduced below.

REGISTER LISTENER

POST /hub

Description

Sets the communication endpoint address the service instance must use to deliver information about its health state, execution state, failures and metrics. Subsequent POST calls will be rejected by the service if it does not support multiple listeners. In this case DELETE /api/hub/{id} must be called before an endpoint can be created again.

Behavior

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 409 if request is not successful.

Usage Samples

Here's an example of a request for registering a listener.

Request
POST /api/hub Accept: application/json {"callback": "http://in.listener.com"}
Response
201 Content-Type: application/json Location: /api/hub/42 {"id":"42","callback":"http://in.listener.com","query":null}

UNREGISTER LISTENER

DELETE /hub/{id}

Description

Clears the communication endpoint address that was set by creating the Hub..

Behavior

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 404 if the resource is not found.

Usage Samples

Here's an example of a request for un-registering a listener.

Request
DELETE /api/hub/42 Accept: application/json
Response
204

PUBLISH EVENT TO LISTENER

POST /client/listener

Description

Clears the communication endpoint address that was set by creating the Hub.

Provides to a registered listener the description of the event that was raised. The /client/listener url is the callback url passed when registering the listener.

Behavior

Returns HTTP/1.1 status code 201 if the service is able to set the configuration.

Usage Samples

Here's an example of a notification received by the listener. In this example “EVENT TYPE” should be replaced by one of the notification types supported by this API (see Notification resources Models section) and EVENT BODY refers to the data structure of the given notification type.

Request

POST /client/listener
Accept: application/json
 	
{
"event": {
EVENT BODY
},
"eventType": "EVENT_TYPE"
}

Response
201

For detailed examples on the general TM Forum notification mechanism, see the TMF REST Design Guidelines.

ACKNOWLEDGMENTS

VERSION HISTORY

Version Number	Date	Modified by	Description
Version 1.0.0	15/04/2016	Pierre Gauthier TM Forum pgauthier@tmforum.org Mariano Belaunde Orange mariano.belaunde@orange.com	Final version
Version 1.0.1	14/06/2016	Alicja Kawecki TM Forum	Updated cover; minor formatting/style corrections prior to publishing for Fx16

RELEASE HISTORY

Release Number	Date	Release led by:	Description
Release 1.0	15/04/2016	Pierre Gauthier TM Forum pgauthier@tmforum.org Mariano Belaunde Orange mariano.belaunde@orange.com	First Release of the Document. Generated from the API Data Model.

CONTRIBUTORS TO DOCUMENT

Veronique Mauneau	Orange
Jean-Luc Tymen	Orange
Mariano Belaunde	Orange
Elaine Haher	Ericsson

August-Wilhelm Jagau	Ericsson
Liuyiling (Sammy)	Huawei
Sunruinan	Huawei
Jiang Yisong	Huawei
George Glass	BT
Pierre Gauthier	TM Forum
Andreas Polz	Infonova
Takayuki Nakamura	NTT