ARCHITECTURE OF THE OCDS OPEN SOURCE CENTRAL UNIT FOR NETWORKING DIGITAL PUBLIC PROCUREMENT SYSTEM

Ver. 4.0 (18.02.2020)

Contract reference: C36346/520/38928
Deliverable 6: Final version of Open Source OCDS-based architecture for electronic public procurement platform

© Prepared under technical cooperation for EBRD UNCITRAL Public Procurement Initiative
by **uStudio LLC** Kiev, Ukraine

LIST OF ABBREVIATIONS

AC	Awarded Contract
API	Application Programming Interface
BPE	Business Process Engine
CA	Contracting Authority
CAN	Contract Award Notice
CDU	Central database unit
CN	Contract Notice
CPV	Common Procurement Vocabulary
CQRS	Command-query responsibility segregation
CRUD	Create, Read, Update, and Delete
DA	Direct Award
DB	Database
DbC	Design by contract
DTO	Data Transfer Objects
EC	European Commission
EDA	Exploratory Data Analysis
EI	Expenditure Item
EO	Economic Operator
EV	Evaluation
FA	Framework Agreement
FS	Funding Source
ISO	International Organization for Standardization
NEPPs	National Electronic Procurement Platform
OCDS	Open Contracting Data Standard
PE	Procuring Entity
PIN	Prior Information Notice
PN	Periodic Notice

PS	Pre-selection
SOA	Service Oriented Architecture
UNCITRAL	The United Nations Commission on International Trade Law
UI	User Interface

TABLE OF CONTENTS

1. Introduction

The current document describes the solution architecture for the OCDS Open Source Central Unit of an electronic public procurement system.

This document provides the description of the software architecture of the system, including:

- technologies to be used to implement the required functionalities;
- structure of the information system identifying each of the subsystems;
- references to software architecture standards and models;
- description of the interactions amongst the different subsystems.

This document cannot be read as a standalone description of an eProcurement system. It rather provides a general approach towards an OCDS based architecture, which shall be complemented by an actual definition document (technical documentation) reflecting how this architecture was developed and implemented in a specific context. Such document will be adapted to specific user needs and specific tasks and activities that are not described in this architecture document.

This system architecture follows an end-to-end approach towards digital public procurement. Therefore, it covers the complete public procurement life cycle, from budgeting and planning to contract management. The architectural implementation of the public pgetrocurement process is provided in the Business Process Engine section.

The document is structured as follows: firstly, an overview of the architecture approach and the system key features and development principles is described, including a high level solution architecture of all system components. Secondly, the Data Model to be implemented following OCDS is detailed, along with building blocks, definition of data components and codelists. Thirdly, the document describes the Business Process Engine, which is the logical layer responsible for managing and executing the business processes. Fourthly, the public API that provides an access point to the system for external users and systems is defined. Lastly, the coverage section provides a complete overview of all the procurement methods that are covered under this eProcurement system architecture.

1.1. Procurement methods covered

This section provides a complete overview of all the procurement methods that covered under this eProcurement system architecture. A general overview of the workflows is provided within this section. The complete BPMN for the procurement methods prescribed shall be accessed through the links indicated in the following subsections.

- Single-stage competitive procedures (parallel and sequential)
  - Small value procedure (simplified open tender)
  - Micro value procedure based on electronic reverse auction
  - EU GPA Open Tender
  - Price Quotation without eCatalogue
- Single-stage non-competitive procedures
  - Negotiated procedures
  - Direct awards
- Multi-stage competitive procedures
  - Restricted Tender
  - Design Contest
  - Competitive Dialog
- Multi-stage repetitive competitive procedures
  - Framework Agreement / Multiple suppliers with direct call offs
  - Framework Agreement / Multiple suppliers with mini-competitions for call-offs
  - Framework Agreement / Multiple suppliers with mini-competitions
  - Dynamic Purchasing System

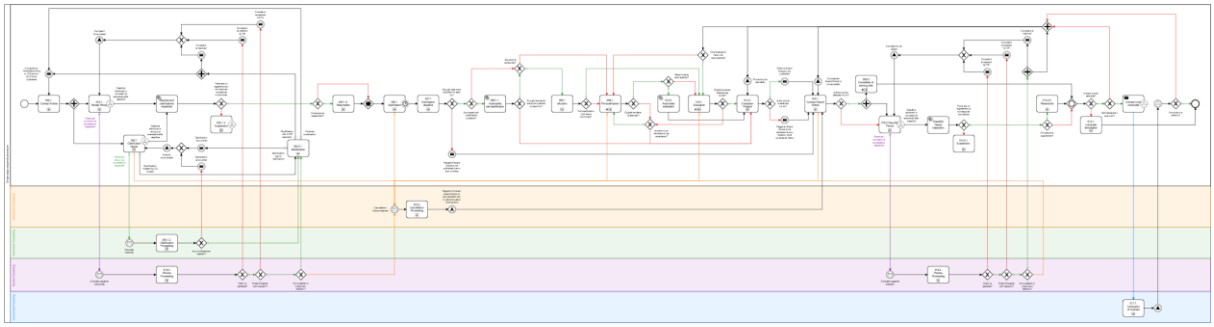
1.1.1. Single-stage competitive procedure

- Small value procedure (simplified open tender)
- EU GPA Open Tender
- Price Quotation without eCatalogue

```
"procurementMethod":                                                    "open"
"procurementMethodDetails": "microValue", "smallValue", "rfq", "openTender"
```

Sequential Type

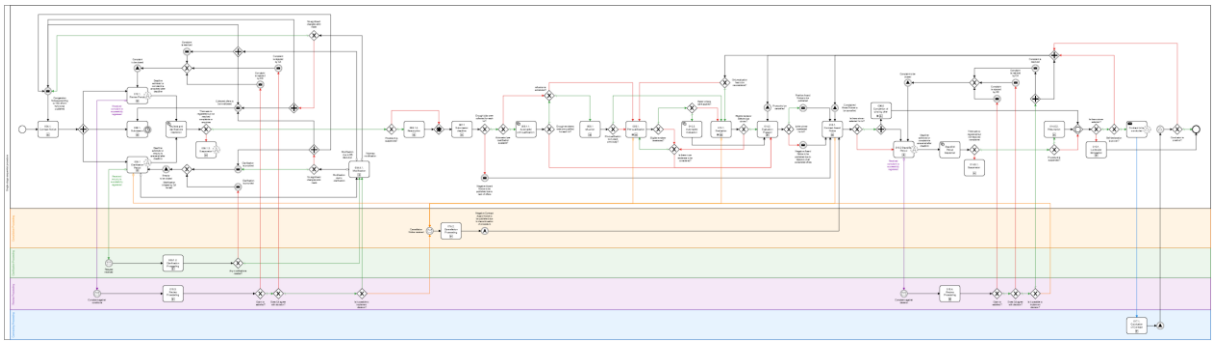
The procurement process is carried out in the following sequence:



Business Process Diagram (go to http://bit.ly/2J13Vf6)

Parallel Type

The procurement process is carried out in the following sequence:



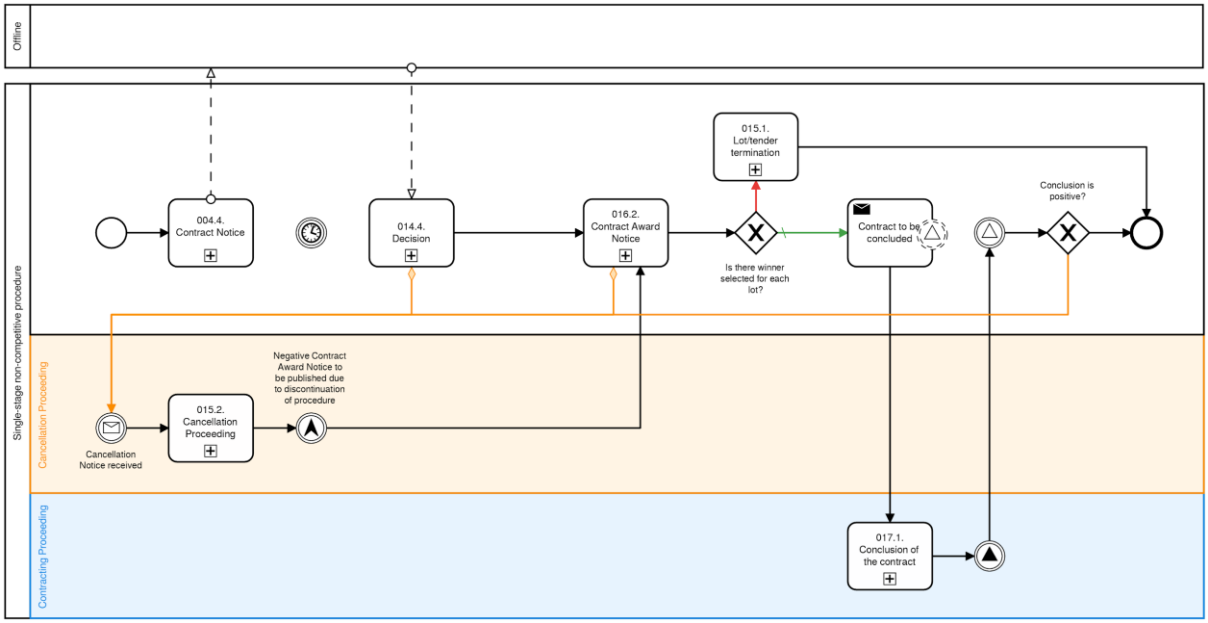
Business Process Diagram (go to http://bit.ly/2Y5Hclj)

1.1.2. Single-stage non-competitive procedure

- Negotiated procedures
- Direct awards

```
“procurementMethod”:                                                    "selective"
“procurementMethodDetails”: "negotiatedProcedure", "directAward"
```

The procurement process is carried out in the following sequence:



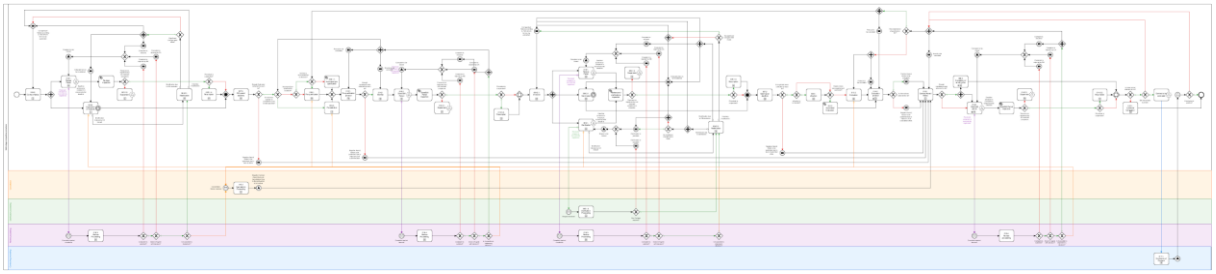
Business Process Diagram (go to http://bit.ly/2ZWVwOR)

1.1.3. Multi-stage competitive procedures

- Restricted Tender
- Design Contest
- Competitive Dialog

```
"procurementMethod":                                                    "open"
"procurementMethodDetails": "restrictedTender", "designContest", "competitiveDialog"
```

The procurement process is carried out in the following sequence:



Business Process Diagram (go to http://bit.ly/2ZZqJB9)

1.1.4. Multi-stage repetitive procedures

Multi-stage repetitive procedure - Framework agreements set out the general terms under which specific purchases ("call offs") can be made under the agreement. The purpose of using a framework is to enable contracting authorities to award individual contracts without going through a full EU procurement process each time.

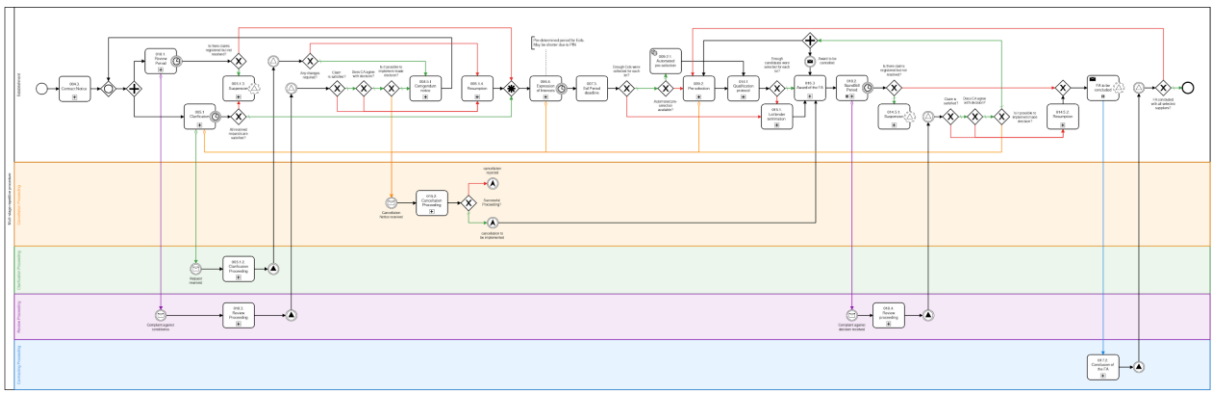
```
"procurementMethod":                                                    "open"
"procurementMethodDetails": "frameworkAgreement", "dynamicPurchasingSystem"
```

The procurement process is splitted into (at least) two stages:

1. Framework establishment
2. Framework execution

Establishment

Part for isolated establishment (focused only on selection of participants for FA execution) is carried out in the following sequence:



Business Process Diagram (go to https://bit.ly/2Vz3afM)

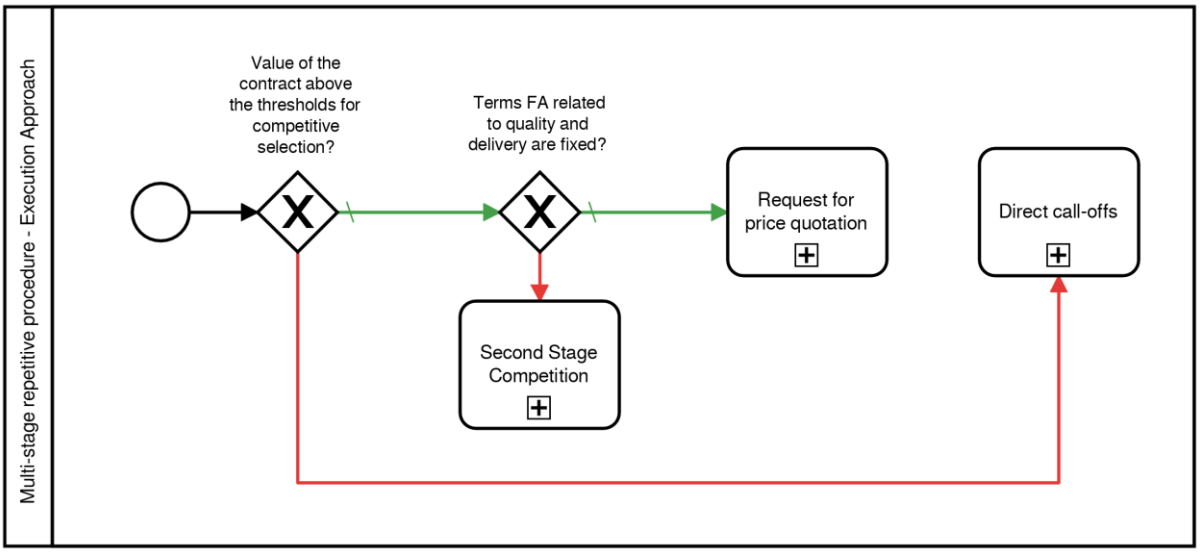
Execution

A framework agreement may be established with one provider ("single sulier frameworks") or with more than one ("multiple suppliers frameworks"). Therefore there are four combinations of process available:

- Framework Agreement / Single supplier with direct call-offs
- Framework Agreement / Multiple suppliers with direct call offs
- Framework Agreement / Multiple suppliers with mini-competitions for call-offs
- Framework Agreement / Multiple suppliers with second-stage competitions

As shown above, where the framework is a multiple suppliers framework, the contracting authority has several options when awarding call off contracts:

- Where the terms of the framework are sufficiently precise to cover the particular call off, the contract may be awarded without reopening competition (i.e. a “direct call off”).
- Where the terms laid down in the framework are not precise enough or complete for the particular call off, a further competition (i.e. a “mini competition”) should be held with all those suppliers within the frameworks capable of meeting the particular need.
- In addition, there is a third option of calling off a contract via a combination of both direct award and mini-competition routes.

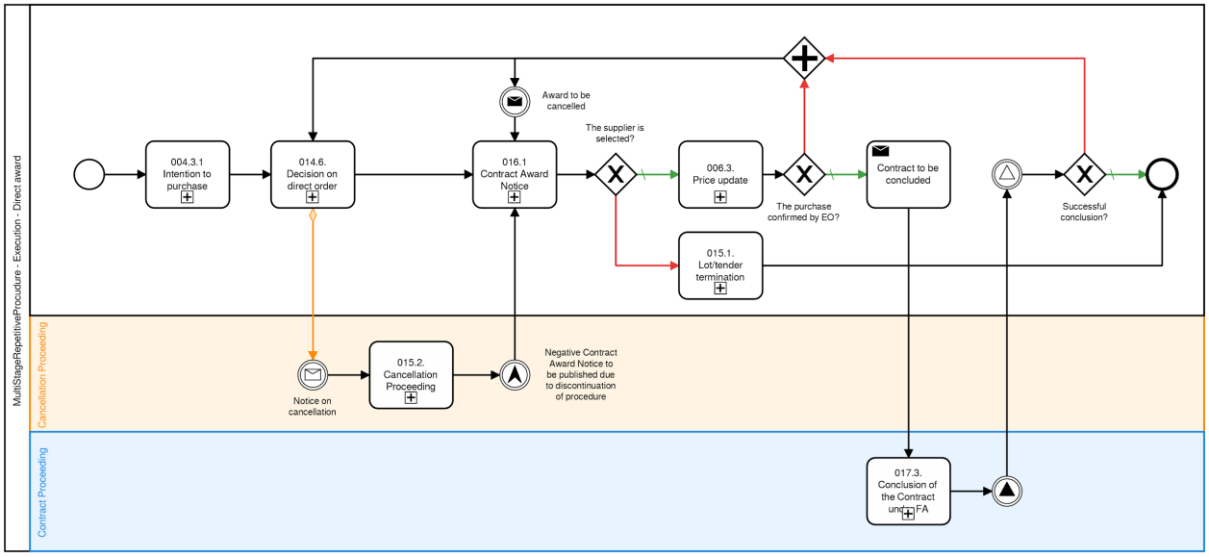


Business Process Diagram (go to http://bit.ly/2Gqbloo)

Therefore, depending on CAs strategy as well as the value of the purchase from FA, execution part is carried out in accordance to one of the following sequences

Direct award

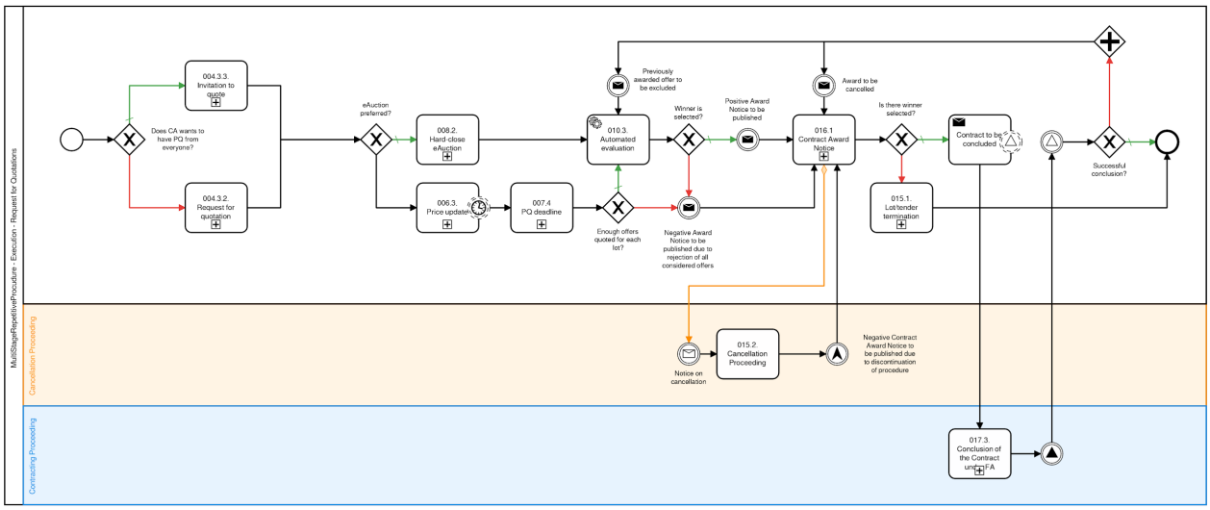
As set out above, where the terms of the framework are sufficiently precise to cover the particular call off, a direct call off may be made. If the framework agreement sets out all the terms governing the provision of the works, services and/or supplies concerned and all the objective conditions that are required to make a decision for award of the specific contract, then awarding the contract without reopening competition amongst the parties to the framework agreement is possible. In this instance, the choice of provider must be based on the objective criteria laid out in the procurement docs.



Business Process Diagram (go to http://bit.ly/2UotAiY)

Request for Price Quotation

Where the terms laid down in the framework agreement are not precise or complete enough for the particular call off, a mini competition should be held with all those suppliers within the framework arrangements capable of meeting the particular need. This does not mean that basic terms can be renegotiated or that the specification used in setting up the framework can be substantively changed. Basic terms can be refined or supplemented to reflect particular circumstances that apply to the call off where these were not and could not be provided for when the framework agreement was established.

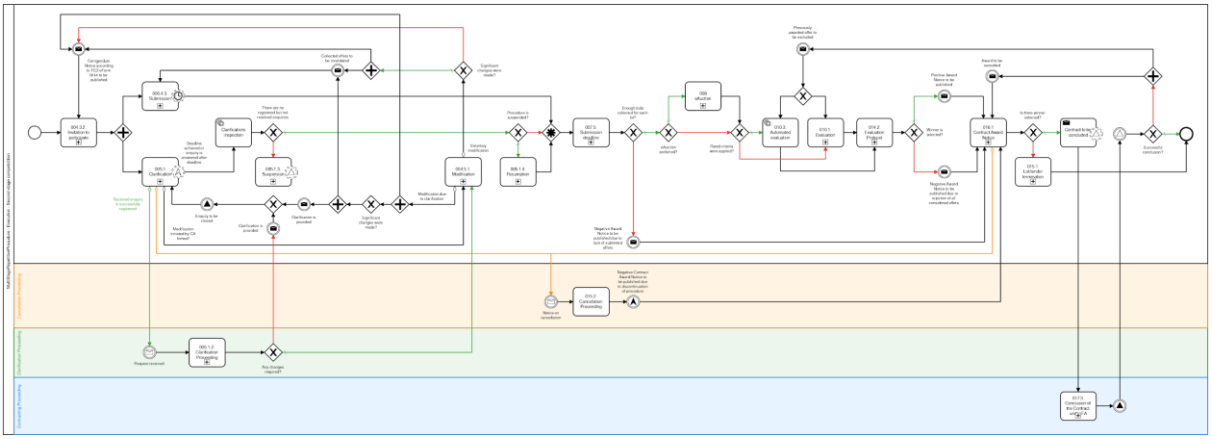


Business Process Diagram (go to http://bit.ly/2XuJFFP)

Second-stage competition

Combination of direct award and mini-competition can be can be used provided that:

- the procurement documents state that this route may be used
- the framework sets out all the necessary terms governing provision of the work/supply/service being procured
- objective criteria are set out in the procurement documents for the framework agreement and are applied to determine whether a particular call off is directly awarded or goes on to a mini- competition
- the procurement documents specify which terms may be subject to the reopening of competition. The guidance gives this example.



Business Process Diagram (go to http://bit.ly/2Vf988A)

1.2. General approach

To ensure the attainment of the objectives set for the IT System, the following principles shall be considered while designing, developing and implementing the Architecture of the OCDS open source central unit of the electronic public procurement system.

1.2.1. Multi-platform networking model

Any system based on this architecture will consists of a CDU that communicates in real time with several NEPPs and external information systems (eGov services) that ensure complete digitization of public procurement procedures. CDU consolidates and distributes all data entered and generated by NEPPs, and performs all transactions, processes and functionalities of the lifecycle and the NEPPs are serve as a user-interface - frolm registration of user account, through UI for all transactions processes and functionalities of lifecycle, up to ability of end-users to use needed eGov Services.

End-to-end public procurement processes

The functionalities of the system based on this architecture from CDU perspective will cover the entire procurement process, from procurement planning to billing and public procurement payments. Online tools will make the system easier for all end-users and will generate process improvements and significant benefits.

Interoperability with eGov Services

This architecture designed to ensure interoperability with existing and future eGovernment services and local national public registers. This approach guarantees a modern and advanced eProcurement system with comprehensive functionalities, but without presenting huge risk or expense since eGovernment services have already been significantly developed in most countries.

Open source, open data and open contracting data standards.

This architecture designed for a stack of open-source technologies to foster transparency and accountability of public procurement by incorporating open data concept and Advanced Open Contracting Data Standards. Openness of data shall help building citizens' and economic operators' trust in the Government, in efficiency of public procurement management and discourage any corrupt practices.

Split-level architecture

Involves independent design of systems' components in compliance with interface standards between levels, which facilitates future incorporation of new services, evolution of the different functionalities and, in general, scalability of the system.

Reliable data

Any data shall be entered into the system through authorised and authenticated channels only.

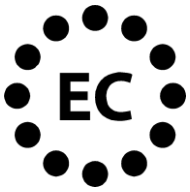
Extensibility

Possibility to expand and supplement the system based on this Architecture with new functions or improve the existing ones.

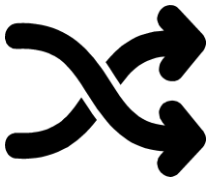
Additional features

The following key features are applied for the development of the architecture of an Open Source Central Unit. These ensure that the architecture can be reusable in other context and implementing different technologies while guaranteeing the interoperability with other systems and being compliant with main standard regulations in public procurement.

Based on United Nations Commission on International Trade Law on Public Procurement (UNCITRAL) (2011)



Complies with the European legislation on Public Procurement (e.g. 2004/17/EC, 2004/18/EC and their amendments)

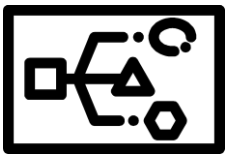


Interoperable with existing official resources constituting the EU Single Market of Public Procurement

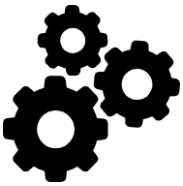
Maximum standardization of data according to international and EU standards: ISO, OCDS, EDI, CEN, etc

Fully open-sourced stack of frameworks of multi-paradigmatic development tools and databases services,

Complete independence from the hardware platforms and/or structures of data-centers or virtual clouds



Business-processes and workflows that are configured from independent parameterized blocks



Reconfigurable in attend to the needs of CAs in a flexible, efficient and customised manner for local or international level

1.2.2. eProcurement.Systems API

The eProcurement.Systems API is the only interface to Central Business Process Engine that is core unit of eProcurement.Systems infrastructure. The eProcurement.Systems API is a CQRS interface that provides programmatic access to BPE of the systems. It provides predictable URLs for accessing resources, and uses built-in HTTP features to receive commands and return responses. This makes it easy to communicate with.

The API accepts JSON or form-encoded content in requests. It returns JSON content in all of its responses, including errors. Only the UTF-8 character encoding is supported for both requests and responses.

2. Design and development principles

This section describes the design and development methodologies followed for the architecture definition of the OCDS Open Source Central Unit of the electronic public procurement system. The general approach on design of architecture for the eProcurement system involves the Service Oriented Architecture (SOA) and Exploratory Data Analysis (EDA) architecture. These two approaches combined provide a richer, more robust model by leveraging previously unknown causal relationships in order to build a new event pattern. This new business intelligence pattern triggers further automated processing that adds exponential value to the enterprise by injecting value-added information into the recognized pattern which could not have been previously achieved.

2.1 Architectural approach

Event-driven SOA that combines the intelligence and proactiveness of EDA with the organizational capabilities found in service offerings.

2.1.1 Service-oriented architecture

SOA is a style of software design where services are provided to the other components by application components, through a communication protocol over a network. The basic principles of service-oriented architecture are independent of vendors, products and technologies. A service is a discrete unit of functionality that can be accessed remotely and acted upon and updated independently, such as retrieving a credit card statement online.

2.1.2 Event-driven architecture

EDA, is a software architecture pattern promoting the production, detection, consumption of, and reaction to events. From a formal perspective, what is produced, published, propagated,

detected or consumed is a (typically asynchronous) message called the event notification, and not the event itself, which is the state change that triggered the message emission. Events do not travel, they just occur. However, the term event is often used metonymically to denote the notification message itself, which may lead to some confusion. Building systems around an event-driven architecture simplifies horizontal scalability in distributed computing models and makes them more resilient to failure. This is because an application state can be copied across multiple parallel snapshots for high-availability. New events can be initiated anywhere, but more importantly propagated across the network of data stores updating each as they arrive. Adding extra nodes becomes easier as well as you can simply take a copy of the application state, feed it a stream of events and run with it. Event-driven architecture can complement SOA because services can be activated by triggers fired on incoming events. This paradigm is particularly useful whenever the sink does not provide any self-contained executive

2.1.3 CQRS architecture style: command-query separation

In traditional architectures, the same data model is used to query and update a database. That's simple and works well for basic CRUD operations. In more complex applications, however, this approach can become unwieldy. For example, on the read side, the application may perform many different queries, returning data transfer objects (Data Transfer Objects or DTOs) with different shapes.

Object mapping can become complicated. On the write side, the model may implement complex validation and business logic. As a result, you can end up with an overly complex model that does too much. Another potential problem is that read and write workloads are often asymmetrical, with very different performance and scale requirements.

Command and Query Responsibility Segregation (CQRS) is an architecture which addresses these problems by separating reads and writes into separate models, using commands to update data, and queries to read data.

- Commands should be task based, rather than data centric. Commands may be placed on a queue for asynchronous processing, rather than being processed synchronously.
- Queries never modify the database. A query returns a DTO that does not encapsulate any knowledge.

Compared to the single data model used in CRUD-based systems, the use of separate query and update models for the data in CQRS-based systems simplifies design and implementation.

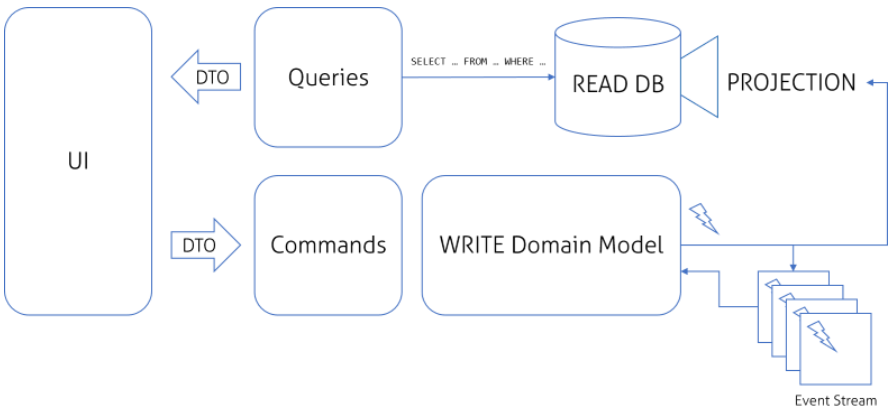


Figure 1. General scheme for CQRS-approach

For greater isolation, the read and the write data could be physically separated. In that case, the read database can use its own data schema that is optimized for queries.

For example, it can store a materialized view of the data, in order to avoid complex joins or mappings. It might even use a different type of data store. If separate read and write databases are used, they must be kept in sync. Typically this is accomplished by having the write model publish an event whenever it updates the DB.

2.1.4 Design by Contract: contract programming approach

Design by contract (DbC), also known as contract programming, programming by contract and design-by-contract programming, is an approach for designing software prescribes that software designers should define formal, precise and verifiable interface specifications for software components, which extend the ordinary definition of abstract data types with preconditions, postconditions and invariants. These specifications are referred to as "contracts", in accordance with a conceptual metaphor with the conditions and obligations of business contracts.

The DbC approach assumes all client components that invoke an operation on a server component will meet the preconditions specified as required for that operation. Where this assumption is considered too risky (as in multi-channel client-server or distributed computing) the opposite "defensive design" approach is taken, meaning that a server component tests (before or while processing a client's request) that all relevant preconditions hold true, and replies with a suitable error message if not.

2.2 Used Patterns

2.2.1 Materialized View pattern

Materialized views (MV) are used to generate pre-populated views of the data in one or more data stores when the data isn't ideally formatted for required query operations. This can help support efficient querying and data extraction, and improve application performance.

Context and problem

When storing data, the priority for developers and data administrators is often focused on how the data is stored, as opposed to how it's read. The chosen storage format is usually closely related to the format of the data, requirements for managing data size and data integrity, and the kind of store in use. For example, when using NoSQL document store, the data is often represented as a series of aggregates, each containing all of the information for that entity. However, this can have a negative effect on queries. When a query only needs a subset of the data from some entities, such as a summary of orders for several customers without all of the order details, it must extract all of the data for the relevant entities in order to obtain the required information.

Solution

To support efficient querying, a common solution is to generate, in advance, a view that materializes the data in a format suited to the required results set. The MV pattern describes generating pre populated views of data in environments where the source data isn't in a suitable format for querying, where generating a suitable query is difficult, or where query performance is poor due to the nature of the data or the data store. These MVs, which only contain data required by a query, allow applications to quickly obtain the information they need. In addition to joining tables or combining data entities, MVs can include the current values of calculated columns or data items, the results of combining values or executing transformations on the data items, and values specified as part of the query. A MV can even be optimized for just a single query. A key point is that a MV and the data it contains is completely disposable because it can be entirely rebuilt from the source data stores.

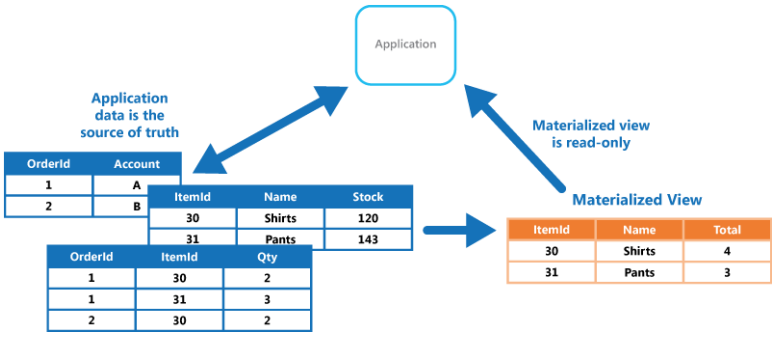


Figure 2a. General scheme of "Materialized View" pattern

A materialized view is never updated directly by an application, and so it's a specialized cache.

When the source data for the view changes, the view must be updated to include the new information. You can schedule this to happen automatically, or when the system detects a change to the original data. In some cases it might be necessary to regenerate the view manually. The figure shows an example of how the Materialized View pattern might be used.

How it works

The following figure shows an example of using the Materialized View pattern to generate a summary of sales. Data in the Order, OrderItem, and Customer tables in separate partitions in an Azure storage account are combined to generate a view containing the total sales value for each product in the Electronics category, along with a count of the number of customers who made purchases of each item.

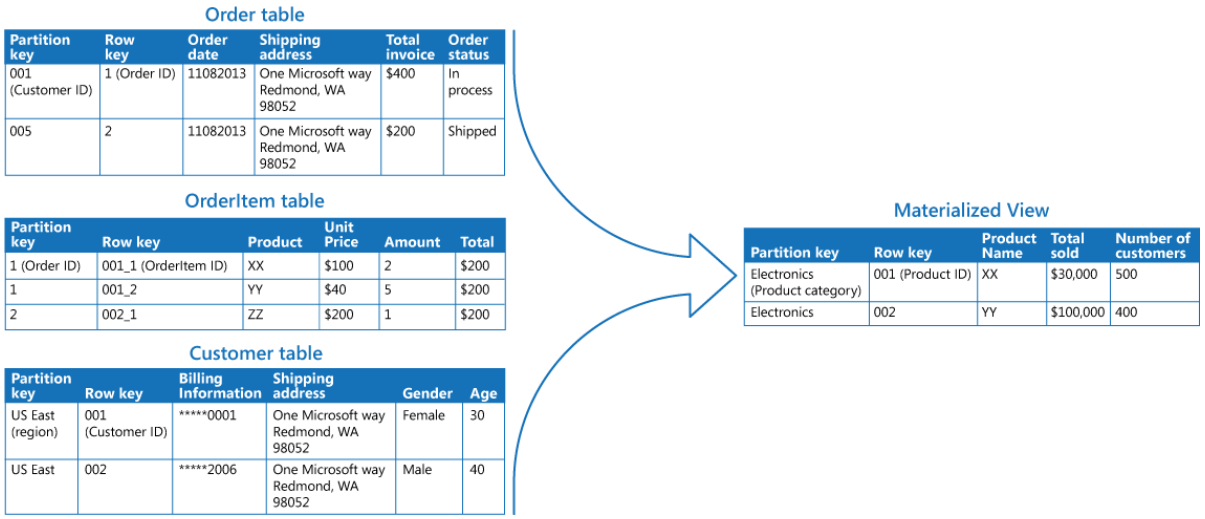


Figure 2b. Tables and materialized view

2.2.2 Event Sourcing pattern

Instead of storing just the current state of the data in a domain, Event Sourcing uses an append-only store to record the full series of actions taken on that data. The store acts as the system of record and can be used to materialize the domain objects. This can simplify tasks in complex domains, by avoiding the need to synchronize the data model and the business domain, while improving performance, scalability, and responsiveness. It can also provide consistency for transactional data, and maintain full audit trails and history that can enable compensating actions.

Context and problem

Most applications work with data, and the typical approach is for the application to maintain the current state of the data by updating it as users work with it. For example, in the traditional Create, Read, Update, and Delete (CRUD) model a typical data process is to read data from the store, make some modifications to it, and update the current state of the data with the new values—often by using transactions that lock the data. The CRUD approach has some limitations:

- CRUD systems perform update operations directly against a data store, which can slow down performance and responsiveness, and limit scalability, due to the processing overhead it requires.
- In a collaborative domain with many concurrent users, data update conflicts are more likely because the update operations take place on a single item of data.
- Unless there's an additional auditing mechanism that records the details of each operation in a separate log, history is lost.

Solution

The Event Sourcing pattern defines an approach to handling operations on data that's driven by a sequence of events, each of which is recorded in an append-only store. Application code sends a series of events that imperatively describe each action that has occurred on the data to the event store, where they're persisted. Each event represents a set of changes to the data. The events are persisted in an event store that acts as the system of record (the authoritative data source) about the current state of the data. The event store typically publishes these events so that consumers can be notified and can handle them if needed. Consumers could, for example, initiate tasks that apply the operations in the events to other systems, or perform any other associated action that's required to complete the operation. Notice that the application code that generates the events is decoupled from the systems that subscribe to the events.

Typical uses of the events published by the event store are to maintain materialized views of entities as actions in the application change them, and for integration with external systems. For example, a system can maintain a materialized view of all customer orders that's used to populate parts of the UI. As the application adds new orders, adds or removes items on the order, and adds shipping information, the events that describe these changes can be handled and used to update the materialized view.

In addition, at any point it's possible for applications to read the history of events, and use it to materialize the current state of an entity by playing back and consuming all the events related to that entity. This can occur on demand to materialize a domain object when handling a request, or through a scheduled task so that the state of the entity can be stored as a materialized view to support the presentation layer.

The figure №3 below shows an overview of the pattern, including some of the options for using the event stream such as creating a materialized view, integrating events with external applications and systems, and replaying events to create projections of the current state of specific entities.

The Event Sourcing pattern provides the following advantages:

- Events are immutable and can be stored using an append-only operation. The user interface, workflow, or process that initiated an event can continue, and tasks that handle the events can run in the background.
- Events are simple objects that describe some action that occurred, together with any associated data required to describe the action represented by the event. Events don't directly update a data store. They're simply recorded for handling at the appropriate time.
- Events typically have meaning for a domain expert, whereas object-relational impedance mismatch can make complex database tables hard to understand. Tables are artificial constructs that represent the current state of the system, not the events that occurred.

- Event sourcing can help prevent concurrent updates from causing conflicts because it avoids the requirement to directly update objects in the data store.
- The append-only storage of events provides an audit trail that can be used to monitor actions taken against a data store, regenerate the current state as materialized views or projections by replaying the events at any time, and assist in testing and debugging the system. In addition, the requirement to use compensating events to cancel changes provides a history of changes that were reversed, which wouldn't be the case if the model simply stored the current state.
- The event store raises events, and tasks perform operations in response to those events. This decoupling of the tasks from the events provides flexibility and extensibility. Tasks know about the type of event and the event data, but not about the operation that triggered the event. In addition, multiple tasks can handle each event. This enables easy integration with other services and systems that only listen for new events raised by the event store.

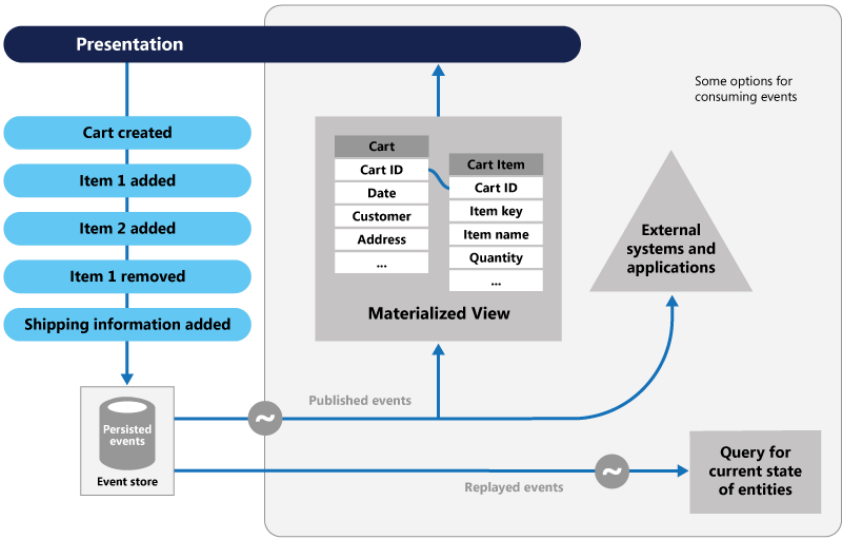


Figure 3. General scheme of “Event sourcing” pattern

3. Hi-level eProcurement solution architecture

3.1 Building blocks

The system consists of following set of components and environments:

3.1.1 NEPPs

The front-end platforms are web platforms that implement eProcurement system features and provide access for CAs and EOs. The NEPPs interact with the CDU via the API and provides suppliers with temporary credentials to access the auction module hosted on the central unit.

3.1.2 Central Unit

- *Business Process Engine* - core systems component responsible for execution of commands received and processes inside the system
- *CDB* - central database, which contains information related to each contracting process
- *Chronograph* - component that controls the time and operates as a scheduler
- *Transport agent* - middleware responsible for transferring data (requests and responses) between the System and third-party services and systems
- *Form Service* - service that allows NEPPs to get actual needed field-set together with existing related information and reference data within one single request to build UI-Form for producing specific type of data-entity by end-user on its side.
- *Auth Service* - self-contained service for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed.
- *Operation Service* - self-contained service responsible for identification of the commands between the CDU (BPE) and the NEPPs.
- *Feed Point* - platform to organize real-time communication between parties (including NEPPs) for transfering a personalized data as a closed feed,
- Public Point - component that displays CDB records in read-only mode. Only information allowed for publication by the current legislation is displayed.

Document Service

Component responsible for registering, uploading, archiving, organising and control of access to all documents (files), enabling long-term preservation of them in digital format and ensuring that they can be easily retrieved without conversions. Service also provides basic document management functionality for receipt, dispatch, storage and retrieval of documents. Component consist of following:

- *Document Repository Management System* - server managing the access to the file repository. It controls the access to the files and reads the files from the repository. Requires Internet Connection.

- *Document Repository* - storage of the files attached to the tender notices, offers, and complaints. Does not require Internet Connection.

Master Data Service

Component is responsible for providing a Data Management, both master and meta. Therefore consumers (such as NEPPs) are able to use centralized dictionary, identification, classification and other kinds of data needed to fulfill data-layer of procurement process according to the data models used (see bellow).

eAuction environment

- *Front-end Auction Servers* - cluster of the front-end servers for the eAuction module, which serves eAuction users: displaying individual user's page; retrieving the current status of the auction from the CDB server, etc.Requires Internet Connection.

- *Auction Public Server* - front-end eAuction server, installed for the observers. It displays the web page for read-only users, so that everybody can watch how an eAuction is running. Current status of the auction is received from the DB auction server. It does not allow making bids.Requires Internet Connection.

- *Back-end Auction Server* - back-end eAuction server. It receives bids uploaded through the front-end auction servers, saves them in the auction DB, and executes the logic of the eAuction module. During the auction, it keeps track of the time allocated to each bidder and it determines the sorting order of the bidders at the end of each round according to the award criteria. Does not require Internet Connection.

- *Auction DB Management Server* - back-end server, which provides correlation between the back-end auction server and the auction DB.Does not require an Internet Connection.

- *Auction DB* - auction database, which contains information about the auctions in progress and bids provided by every participant.Does not require an Internet Connection.

Web portal

Website that presents procurement information in compliance with legislation, from the CDB through the Portal's Public. Contains general information pages as well. Requires Internet Connection. Web portal is described in the functional document "EBRD_MTender_Technical Specifications_v16.0.docx", in section 4.1.

3.2 Architecture diagram

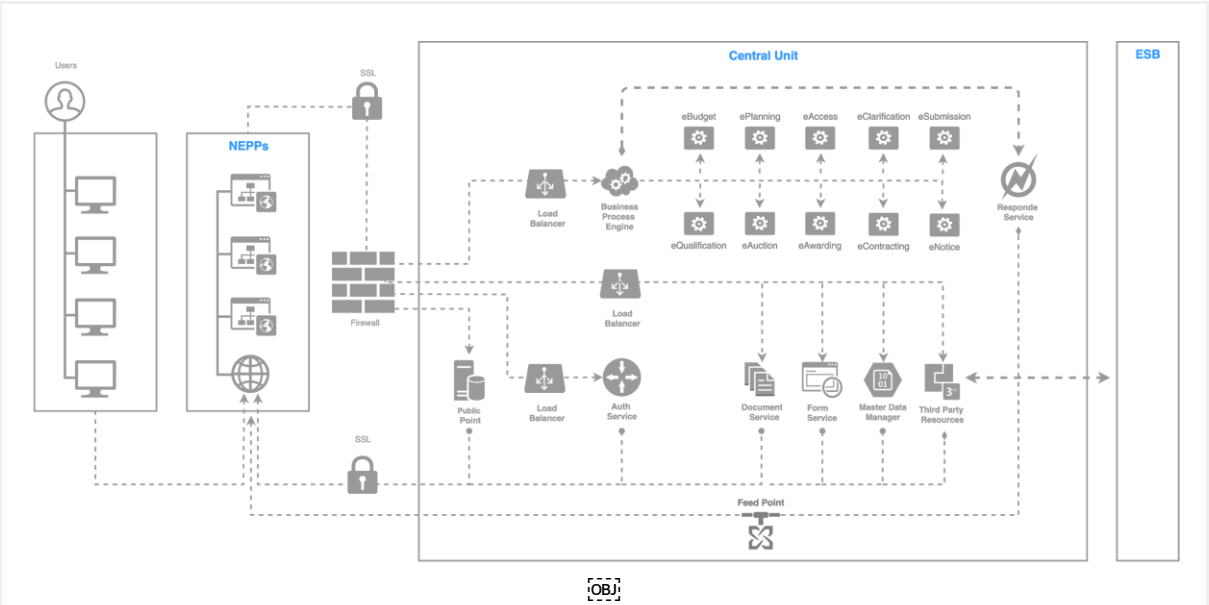


Figure 4. High-level diagram of the systems' architecture

4 Data Standard

4.1 Open Contracting Data Standard 1.1+

Data standard is modelled along the Open Contracting Standard[1] with extensions in areas that were not covered by it. OCDS is an open data standard for publication of structured information on all stages of a contracting process. The Open Contracting Data Standard is maintained using JSON Schema. The main sections and common objects used in the schema are introduced in the getting started section.

4.1.1 OCDS structure

The OCDS provides a detailed specification of the fields and data structures to use when publishing open data about contracting processes. The OCDS concept of a release can be broadly equated to the concept of a 'notice' UNCITRAL regulations.

[1] http://standard.open-contracting.org/latest/en/

The core OCDS schema is designed to support global interoperability, and to include a common core of data fields required by a wide range of use cases. The OCDS schema can be extended with definitions for new data structures to meet the needs of particular publishers.

A typical OCDS release consists of Package, Release and one or more Sections. This documentation includes Package and Release within this Structure subheading. Sections, such as Tender, Award and Contract are explained within the Sections sub-heading. Objects are explained within the Building Blocks and Extensions Building Blocks.

4.1.2 Building blocks

Building blocks are OCDS data structures that are smaller than most sections and are coherent blocks of data in their own right. It should be possible to read a building blocks data in isolation and for it to make sense. Where blocks have other building blocks nested within them the documentation shows how those are structured and then provides an example of the data that describes the building blocks together.

The majority of OCDS data is held within a release structure. One or more releases can be published within a release package. Releases are made up of a number of sections, including the following.

Parties

Each of the parties (organizations or other participants) referenced in a release must be included in the parties section: information on the parties who are involved in the contracting process and their roles, e.g. buyer, procuring entity, supplier etc. Organization references elsewhere in the schema are used to refer back to this entries in this list.

Planning

The planning section is used to describe the financial background to a contracting process. This include details of the budget from which funds are drawn or related budget-projects for this contracting process.

Budget

This section contains information about the budget line and/or associated projects, through which this contracting process is funded. It used to cross-reference to more detailed information held using a Budget Data Package (Expenditure Item and/or Budget Project) or where no linked Budget Data Package is available, to provide enough information to allow cross-reference with another published source of budget and project information.

Tender

The tender section includes details of the announcement that an organization intends to source some particular goods, works or services, and to establish one or more contract(s) for these.

Bids

The bid section is used to publish summary statistics and individual bid information. Where information is provided on individual bids, these statistics should match those that can be calculated from the bid details array and an array of bids, providing information on the bidders, and where applicable, bid status, bid values and related documents. The extent to which this information can be disclosed varies from jurisdiction to jurisdiction.

Prequalification

Data regarding prequalification process - limiting the number of qualified bidders by reviewing each bidder's qualifications against a set of criteria.

Awards

An award for the given procurement. There may be more than one award per contracting process e.g. because the contract is split amongst different providers, or because it is a standing offer.

Contract

The contract section is used to provide details of contracts that have been entered into. Every contract needs to have a related award, linked via the awardID property. This is because supplier information is contained within the 'award'. The framework contract details below help illustrate the reasons for this.

> ### Extending the standard
>
> The OCDS release schema can be extended with definitions for new fields and data structures to meet the needs of particular communities of data publishers. This document also provides key information on using an extended release schema for the UNCITRAL.

4.1.3 Procurement Process Stages

Depending on the type of procedure (tender.procurementMethodDetails) different stages can be used. In general, there are 4 main stages of the procurement process, from which the competitive phase of the purchasing process can consist (in different combinations and / or sequences):

Pre-selection

This is a period during which interested parties can clarify any details of object of future procurement with CA and with respond to an initial contract notice are submits a short statement of information about the organization according to instructions or a questionnaire. These form the basis of a qualification submission that such parties must make to demonstrate their ability to implement the project. CA selects several candidates to be invited to the next stage. This stage relates specifically to the selection of competent tenderers, prior to the issue of invitations to tender with a main purpose to select those potential suppliers, whose qualifications and experience would minimise the risk of non-performance.

Pre-selection Stage (PS) data-model

The *pre-selection* of participants of future stages usually the first active stage of multi-stage procedures (restricted tender or any other). Therefore, the core of data-model of the pre-selection stage is the contractNotice and described as Release Package.

Since the stage is active (meaning that there are user actions form CA and EOs expected) in its process there are data describing its course and result generated by Users Actions. Such data are described by objects complementing the initial data model of the stage (see description below).

Pre-Qualification

During 'Pre-qualification' stage all invited (pre-selected) parties can also clarify any additional details of object of future procurement with CA and with respond to an invitation are submits a technical offer according to instructions or a questionnaire. These form the basis of a pre-qualification submission that such parties must make to demonstrate benefits and superiorities of theirs offers. CA selects several candidates to be invited to the next stage.

The pre-qualification process itself consists of the examination and evaluation of the technical part of received offers according to specifications and requirements as defined in the tender documentation, analyse their benefit and determine their superiority. Pre-qualification is the basis for the subsequent selection of the offer that provides the best value for money for the CA.

Pre-qualification Stage (PQ) data-model

Note that pre-qualification of participants of future stages may goes either the first (and even include functionality of PS) or one of the subsequent stages of a multi-stage tender as stand-alone stage

In case of PQ is a first active stage of multi-stage tender, the core of data-model of such 'PQ' is equal to PS data-model described above.

If PQ used in a Procurement Process as subsequent stage, an extension to initial data-model is used to describe a list of invited (usually - pre-selected) participants. This model extension is equal to model of definition of disclosed bids described below. All included 'bids' will be indicated with initial status 'invited' that means that this is a "drafts" of future possible technical bids, based on the results of previous stage (in case of PQ usually it's a PS). Also all tenderers who passed previous stage and invited to participate in PQ will be included in a 'parties' section of definition of such PQ as 'tenderers'. Since the stage is active (meaning that there are user actions form CA and EOs expected) in its process there are data describing its course and result generated by Users Actions. Such data are described by objects complementing the initial data model of the stage (see description below).

Given that the process of selection of qualified suppliers at the PQ in repetitive procedures (FA, DPS) leads to the obligation to Contract Award Notices, the data-model of this stage is also could be expanded with one more section - 'contracts' that includes auto-generated CANs. This model extension is equal to model of definition of Contract Award Notices described below.

Evaluation

During 'Evaluation' stage all invited (pre-selected and/or pre-qualified) parties can also clarify any additional details of object of future procurement with CA and with respond to an invitation are submits a financial offer according to instructions or a questionnaire. CA selects several candidates to be invited to the next stage. Financial evaluation of the proposals involves consideration the offers with the financial and commercial requirements stipulated in the tender documentation. It is also important to identify whether they are within the available budget for the project.

Evaluation Stage (EV) data-model

Note that Evaluation stage may go either the first (and include one or both previous steps: PS and PQ) or one of the subsequent stages of a multi-stage tender as stand-alone stage.

In case of EV is a first active stage of multi-stage tender, the core of data-model of such pre-qualification stage is equal to cnPreselectionStage data-model described above. If EV used in a Procurement Process as subsequent stage, an extension to initial data-model is used to describe a list of invited (usually - pre-qualified) participants. This model extension is equal to model of definition of disclosed bids described below. All included 'bids' will be indicated with initial status 'invited' that means that this is a "drafts" of future possible financial bids, based on the results of previous stage (in case of EV usually it's a PQ). Also all tenderers who passed previous stage and invited to participate in EV will be included in a 'parties' section of definition of such EV as 'tenderers'.

Since the stage is active (meaning that there are user actions form CA and EOs expected) in its process there are data describing its course and result generated by Users Actions. Such data are described by objects complementing the initial data model of the stage (see description below).

Given that the process of selecting a winner at the EV leads to the obligation to Contract Award Notices, the data-model of this stage is also expanded with one more section - 'contracts' that includes auto-generated CANs. This model extension is equal to model of definition of Contract Award Notices described below.

4.1.4 Data Transfer Approach

Data-releases must be published within a release package, which can contain one or more releases. The release package, modelled on the Data Package protocol, provides metadata about the release(s) it contains, the publisher, and data licensing information.

The following points describe some information about the releases and records:

- Releases are immutable, records are updated over the life of the contracting processes.
- Releases are point in time updates, records show the most up to date information.
- A single contracting process can have many releases but must have only one record.
- Records either store a history of changes **or** links to individual releases.
- The OCID remains consistent across all releases and records whilst the release ID changes with each release.
- The value of persistent releases for spotting red flags.
- Releases can **repeat or update** information from previous releases.
- Release tags help to identify what stage of the contracting process a release is about

4.1.4.1 Record Package

The Record Package contains a list of records along with publishing metadata. The records pull together all the releases under a single OCID and compile them into the latest version of the information along with the history of any data changes.

Record Package Schema on GitHub/Open-Contracting

Schema

uri		string	auto-generated
The URI of this package that identifies it uniquely in the world			
version		string	auto-generated
The version of the OCDS schema used in this package			
extensions	array	string	auto-generated
An array of OCDS extensions used in this package			
publisher	object		auto-generated
Information to uniquely identify the publisher of this package See Publisher			
license		string (uri),null	auto-generated

A link to the license that applies to the data in this data package.			
publicationPolicy		string (uri),null	auto-generated
A link to a document describing the publishers publication policy.			
publishedDate		string (date-time)	auto-generated
The date that this package was published.			
packages	array	string (uri)	auto-generated
A list of URIs of all the release packages that were used to create this record package			
records	array	object	auto-generated
The records for this data package			

4.1.4.2 Record

The record (Release Package) schema describes the container document for publishing releases.

Schema

ocid		string	auto-generated
A unique identifier that identifies the unique part of multi-stage Open Contracting Process.			
releases	array	string (uri),null	auto-generated
A list of objects that identify the release associated with this OCID. The releases are sorted into date order in the array, from oldest to newest.			
compiledRelease	array	object	auto-generated
This is the latest version of all the contracting data, provided as an array of releases			
versionedRelease		string (uri),null	auto-generated
This contains the history of the data with all versions of the information and the release they came from			

4.1.4.3 Linked releases

Information to uniquely identify the release

Schema

url		string (uri),null	auto-generated
The URL of the release which contains the URL of the package			
date		string (date-time),null	auto-generated
The date this information was first released, or published.			
tag	enum	string	auto-generated
One value from the releaseTag codelist. Tags may be used to filter release and to understand the kind of information that a release might contain. The tag should match the tag in the release. This provides additional context when reviewing a record to see what types of releases are included for this ocid.			

4.1.4.4 Release Package

All information about a current stage of specific part of contracting process is described with release.

Release Package Schema on GitHub/Open-Contracting

Schema

ocid		string	auto-generated
A unique identifier that identifies the unique part of multi-stage Open Contracting Process.			
id		string	auto-generated
An identifier for this particular released information			

date		string	auto-generated
The date this information was first released, or published.			
tag	array	string	auto-generated
One or more values from the releaseTag codelist. Used to understand the kind of that a release might contain.			
initiationType		string	auto-generated
String specifying the type of initiation process used for this contract, taken from the initiationType codelist.			
parties	array	object	auto-generated
Information on the Organizations who are involved in the process and their roles. See Organization			
buyer		object	auto-generated
The buyer is the entity whose budget will be used to purchase the goods. See Organization reference			
planning		object	optional
Information from the planning phase of the contracting process. See Planning			
tender		object	optional
The activities undertaken in order to enter into a contract. See Tender			
bids	array	object	auto-generated
The bid section is used to publish summary statistics, and where applicable, individual bid information. See Bids			
awards	array	object	auto-generated
Information from the award phase of the contracting process. See Award			
contracts	array	object	auto-generated
Information from the contract creation phase of the procurement process. See Contracts			
relatedProcesses	array	object	auto-generated
A link to a related contracting process in OCDS and the type of relationship See RelatedProcess			
hasPreviousNotice		boolean	optional
True/False field whether this release represents a notice that is connected to a previous notice			
buyerInternalReferenceId		string, null	optional
The buyer's internal reference identifier is an EU specific field.			
purposeOfNotice		object	optional
Details about the purpose of this notice release See PurposeOfNotice			
relatedNotice	array	object	optional
Information that connects a notice with a related notice for the contracting process. See RelatedNotice			

4.1.5 Definitions

For more information about entitities definitions, you can find the extended version of this section in the latest version of the document *Definitions.xlsx*.

From eProcurementSystem.MD:

AcceleratedProcedure

isAcceleratedProcedure	boolean	default:false
A True/False field to indicate whether an accelerated procedure has been used for this procurement		

acceleratedProcedureJustification	string, null	
Justification for using an accelerated procedure		

Address

streetAddress	string	
The street address. For example, 1600 Amphitheatre Pkwy.		
postalCode	string, null	
The postal code. For example, 94043..		
addressDetails	object	
See AddressDetails		

AddressDetails

country	object	
Classification of the country according to one of available schemes See Classification		
locality	object	
Classification of the locality according to one of available schemes See Classification		
region	object	
Classification of the region according to one of available schemes See Classification		

Amendment

date	string, date-time	auto-generated
The date of this amendment.		
rationale	string, null	
An explanation for the amendment		
id	string, null	auto-generated
An identifier for this amendment: often the amendment number		
description	string, null	
A free text, or semi-structured, description of the changes made in this amendment.		
amendsReleaseID	string, null	auto-generated
Provide the values for this contracting process before the amendment was made.		
releaseID	string, null	auto-generated
Provide the values for this contracting process after the amendment was made.		
type	string, null	auto-generated
A type for this amendment taken from a Amendment types codelist		
status	string, null	auto-generated
A status for this amendment taken from a Amendment statuses codelist		
relatesTo	string, null	auto-generated
This ‘amendment’ can be specifically related to one of the lots. Where this is a case, this attribute to be filled as a ‘lot’		
relatedItem	string, null	auto-generated
Where ‘relatesTo:lot’ for this ‘amendment’ a unique id of the related lot shall be reflected by this attribute		

Award

id	string, date-time	auto-generated
The identifier for this award		

suppliers	array	object	auto-generated
See OrganizationReference			
documents	array	object	
All documents and attachments related to the award, including any notices. See Document			
value		string, null	auto-generated
The total value of this award See Value			
items	array	object	auto-generated
The goods and services awarded in this award, broken into line items See Item			
status		string, null	auto-generated
The current status of the award drawn from the AwardStatuses codelist			
statusDetails		string, date-time	auto-generated
Additional details of an award status according to AwardStatusDetails codelist			
date		string, null	auto-generated
The date of the contract award. This is usually the date on which a decision to award was made.			
relatedLots	array	string, null	auto-generated
If this award relates to one or more specific lots, provide the identifier(s) of the related lot(s) here.			
relatedBid		string, null	auto-generated
The identifier of the bid related to this award			
requirementResponces	array	object	auto-generated
See RequirementResponse			
weightedValue		object	auto-generated
Total weighted value for this award calculated with a scoring function applied See Value			

BankAccount

description		string	
Any free text description related to the Bank Account			
bankName		string	
The legal name of the bank			
address		object	
Bank address described as standard ‘address’ object See Address			
identifier		object	
The main identifier of the Bank according to one of available Banks schemes See Identifier			
accountIdentification	array	object	
The settlement account according to one of available bank account identification schemes See Classification			
additionalAccountIdentifiers	array	object	
Additional identifiers of bank account (e.g. fiscal code of the bank if applicable) See Identifier			

Bid

id		string, null	auto-generated
A local identifier for this bid			
date		string, date-time	auto-generated
The date when this bid was received.			
status		string, null	auto-generated
The status of the bid, drawn from the bidStatus codelist			
statusDetails		string, null	auto-generated
The status details of the bid, drawn from the bidStatusDetails codelist			
tenderers	array	object	
See OrganizationReference			
value		object	
The total value of the bid See Value			
documents	array	object, null	
All documents and attachments related to the bid and its evaluation. See Document			
relatedLots	array	string	
If this bid relates to one or more specific lots, provide the identifier(s) of the related lot(s) here.			
requirementResponses	array, null	string	
A set of responses by EO against the requirements applied by PE for eligibility check and evaluation See RequirementResponse			

BidStatistic

id	array	string, null	auto-generated
An internal identifier for this statistical item.			
measure		string, null	auto-generated
An item from the bidStatistics codelist for the statistic reported in value.			
date		string, date-time	auto-generated
The date when this statistic was last updated. This is often the closing date of the tender process.			
value		number,null	auto-generated
The value for the measure in question. Total counts should be provided as an integer.			
notes		string, null	auto-generated
Any notes required to understand or interpret the given statistic.			
relatedLot		string, null	auto-generated
If this statistic relates to bids on a particular lot, provide the lot identifier here.			

BudgetSource

budgetBreakdownID	string
ID of part of initial defined budget using BudgetBreakdown.id as a link	
amount	number
The amount of available fundings	

Budget

id		string
An identifier for the budget		
description		string
A short free text description of the budget		
project		string
The name of the project that through which this contracting process is funded (if applicable)		
projectID		string
An external identifier for the project that this contracting process forms part of, or is funded via (if applicable).		
uri		string
A URI pointing directly to a machine-readable record about the budget		
europeanUnionFunding		object
Details where a procurement has partial or full financing from a European funded project See EuropeanUnionFunding		
isEuropeanUnionFunded		boolean
A True or False field to indicate whether this procurement is related to a project and/or programme financed by EU funds.		
budgetAllocation	array	object
Structured information about allocation of fundings under this concluded contract See BudgetAllocation		
budgetSource	array	object
Set of sources of fundings defined for this concluded contract See BudgetSource		
sourceEntity		object
Reference the organization providing the budget See OrganizationReference		
verified		boolean
A True or False field to indicate whether this funds are verified by authorized entity (Treasury)		
verificationDetails		string
The rationale for the verification provided in free text.		
amount		object
Object described the sum of the available funds under this EI See Value		
budgetBreakdown	array	object
Detailed budget breakdown to be expressed, covering multiple budget sources and multiple periods See BudgetBreakdown		

BudgetBreakdown

id		string
An identifier for this particular budget entry (equal to OCID of used FS)		
description		string, null
A short free text description of this budget entry.		
amount		object
The value of the budget line item. See Value		
uri		string
A URI pointing directly to a machine-readable information about this budget entry.		
sourceParty		object

Reference the organization providing the funds under this budget-line See OrganizationReference	
period The period covered by this budget entry. See Period	object

BudgetAllocation

budgetBreakdownID ID of part of initial defined budget using BudgetBreakdown.id as a link	string
period The period on which specific budget is allocated See Period	object
relatedItem Identifier of specific item on which allocated budget is related	string
amount The amount of available fundings	number

BusinessFunction

id ID for this function	string
type This field is used to indicate the code of function according to the BusinessFunctions codelist	string,url
jobTitle Job title name	string,null
period Period of occupation of this business function See Period	object
documents See Document	string,url

Classification

scheme This field is used to indicate the scheme/codelist from which the classification is drawn	string,url
id The classification code drawn from the selected scheme.	string,null
description A textual description or title for the code.	string,null
uri A URI to identify the code	string,url

Coefficient

id The identifier of this coefficient	string
value Requirement value for this coefficient	number, boolean, string, integer
coefficient	number

The value of the coefficient

ContactPoint

name		string
The name of the contact person, department, or contact point, for correspondence relating to this process		
email		string
The e-mail address of the contact point/person		
telephone		string,null
The telephone number of the contact point/person. This should include the international dialling code		
url		string,null
A web address for the contact point/person		

ConfirmationRequest

id		string
The identifier for this request		
title		string,null
Request title		
description		string,null
Request description		
source		string,null
Source of response to the requirements specified in the criterion as one of the available types,		
relatesTo		string,null
The schema element on which this request is related		
relatedItem		string,null
The id of related element		
requestGroups	array	object
A group of actors related to this request. See RequestGroup		
type	enum	string
The way or confirmation according to RequestType codelist		

ConfirmationResponse

id	string
The identifier for this confirmation response	
value	object
See ConfirmationResponseValue	
request	string
The id and of the request on which the response is applicable to	

ConfirmationResponseValue

name	string
The name of Entity whose representative did this conformation	
id	string
The identifier of Entity whose representative did this conformation	
date	string
The date than this confirmation has been done	
relatedPerson	object

Reference to the person who did this confirmation
See PersoneReference

verification		object	
The type and the body of confirmation substance See Verification			

Contract

id		string	auto-generated
The identifier for this contract			
awardID			auto-generated
The award.id against which this contract is being issued.			
extendsContractID		string,null	auto-generated
If this contract extends or amends a previously issued contract, then the contract.id provided here.			
title		string,null	
Contract title			
description		string,null	
Contract description			
status	enum	string,null	auto-generated
The current status of the contract. Drawn from the contractStatus codelist			
statusDetails		string	auto-generated
The details of the status of the contract specified with contract statusDetails codelist			
agreedMetrics	array	object	auto-generated
Any target metrics set out as part of the contract or its terms See Metric			
period		object	
The start and end date for the contract. See Period			
value		object	auto-generated
The total value of this contract. See Value			
confirmationRequests	array	object	auto-generated
The set of confirmations required under this contract See ConfirmationRequest			
confirmationResponses	array	object	
The set of responses of requested confirmations See ConfirmationResponse			
documents	array	object	
All documents and attachments related to the contract, including any notices. See Document			
implementation	array	object	
See Implementation			
relatedProcesses	array	object	auto-generated
Details of the related process can be provided here. See RelatedProcess			
milestones	array	object	
A list of milestones associated with the finalisation of this contract. See Milestone			
amendments	array	object	
See Amendment			

date	date	auto-generated
Date when this contract was issued		

Conversion

id	string
The identifier for this conversion	
description	string,null
Conversion description	
rationale	string
The rationale provided in free text	
coefficients	array
A set of coefficients applicable within this conversion for each value available under related requirement See Coefficient	
relatesTo	enum
The schema element that the conversion relates to	
relatedItem	string
The id of related element	

Criterion

id	string
The identifier for this criterion	
title	string,null
Criterion title	
description	string,null
Criterion description	
source	string,null
Source of response to the requirements specified in the criterion as one of available types	
relatesTo	string,null
The schema element that the criterion judges, evaluates or assesses. E.g. criterion may be defined against items or bidders	
relatedItem	string,null
The id of related element	
requirementGroups array	object
A list of requirement groups for this Criterion See RequirementGroup	

Details

typeOfBuyer	string
The type of buyer taken from the EU's specified list [typeOfBuyer]	
mainGeneralActivity	string
The main general activity of the buyer taken from the EU's specified list mainGeneralActivity codelist	
mainSectoralActivity	string
The activity of the buyer taken from the mainSectoralActivity codelist	
isACentralPurchasingBody	boolean
A true/false field to indicate whether the organization is a central purchasing body	

scale		string
For commercial organization's, is this a micro, Small or Medium or large entity according to the Scale codelist		
gpaProfile		object
See GpaProfile		
bankAccounts	array	object
See BankAccount		
legalForm		object
Classification of the organization according to Classifier of organizational and legal forms of entities of Moldova (MD-CFOJ) See Classification		
mainEconomicActivity	array	object
Identifier of the registered activity according to Classification of the types of economic activity (MD-CTEA) See Classification		
permits	array	object
Information about permits and licenses issued for the organization See Permit		
typeOfSupplier		string
Classification of the supplier according to typeOfSupplier codelist		

Document

id		string
A local, unique identifier for this document.		
documentType		string
A classification of the document described taken from the documentType codelist.		
title		string,null
The document title.		
description		string,null
A short description of the document.		
url		string,url
A direct link to the document or attachment.		
datePublished		string,date-time
The date on which the document was first published.		
dateModified		string,date-time
Date that the document was last modified		
format		string,null
The format of the document taken from the IANA Media Types code list.		
language		string,null
Specifies the language of the linked document using either two-digit ISO 639-1		
relatedLots	array	string,null
If this document relates to a particular lot, provide the identifier(s) of the related lot(s) here.		

EconimicActivityReference

id		string
The identifier of specific economic activity described for EO		
description		string
Human readable description of related activity		

ElectronicWorkflows

useOrdering	boolean	
A True/False field to indicate if electronic ordering will be used. Required by the EU		
usePayment	boolean	
A field to indicate whether the Dynamic Purchasing System may be used by buyers outside the notice.		
acceptInvoicing	boolean	
A True/False field to indicate if electronic invoicing will be accepted. Required by the EU		

ElectronicAuctions

statistics	object	auto-generated
Aggregate statistics about the auctions related to this specific contracting process		
details	array	
An array, providing information for each auction scheduled under this specific contracting process See ElectronicAuctionDetails		

ElectronicAuctionDetails

id	string	auto-generated
A unique identifier for the auction		
relatedLot	string	
If this auction relates to a specific lot, this field contains the lot identifier		
electronicAuctionModalities	array	
The set of specific attributes and values of this auction See ElectronicAuctionModality		
auctionPeriod	object	
Auction period for specific electronic auction		
electronicAuctionProgress	array	
The array describes set of parts of the auction (rounds) and users action (moves) under this parts See ElectronisAuctionRound		
electronicAuctionResult	array	
The array describes a result of specific electronic auction See ElectronicAuctionResult		

ElectronicAuctionModality

eligibleMinimumDifference	object	
The minimal step available for the specific electronic auction See Value		
url	string	auto-generated
The url of this auction for viewers		

ElectronicAuctionResult

relatedBid	string	
The id of tenderers' bid		
value	object	auto-generated
The value for final tenderers' offer submitted under auction See Value		

ElectronicAuctionRound

id	string	

The minimal step available for the specific electronic auction		
period See Period	object	auto-generated
breakdown The minimal step available for the specific electronic auction See ElectronicAuctionRoundDetails	object	

ElectronicAuctionRoundDetails

relatedBid	string	auto-generated
The id of tenderers' bid		
status	object	auto-generated
The status of tenderers' action (move) under specific part of auction (round) from auction moves codelist		
date	date	
The minimal step available for the specific electronic auction		
value	object	
The minimal step available for the specific electronic auction See Value		

EuropeanUnionFunding

projectIdentifier	string,null	
National identifier of the EU project providing partial or full funding		
projectName	string,null	
Name or other national identification of the project providing full or partial funding.		
uri	string(url),string	
Uri of the project providing full or partial funding.		

Enquiry

id	string	auto-generated
A unique identifier for the enquiry.		
date	string	auto-generated
The date the enquiry was received or processed.		
author	object	
See OrganizationReference		
title	string	
The subject line of the question.		
description	string	
The body of the question.		
answer	string	
The answer to this question, when available.		
dateAnswered	string,date-time	auto-generated
The date the answer to the question was provided.		
relatedItem	string	
If this question relates to a specific line-item, this field contains the line-item identifier.		
relatedLot	string	
Where lots are used, if this question relates to a specific lot, this field contains the lot identifier.		

GPA Profile

gpaAnnex	object
Classification of Entity according to e-GPA scheme See Classification	
gpaOrganizationType	object
Classification of Entity according to GPA Types of Organization	

See Classification

gpaThreshold	array	object
See GPAThreshold		

GPAThreshold

mainProcurementCatogory	string
The primary category describing the main object of this tender from the procurementCategory codelist.	
value	object
See Value	

Implementation

transactions	array	object
A list of the spending transactions made (or planned) against this contract See Transaction		
milestones	array	object
As milestones are completed, milestone completions should be documented. See Milestone		
documents	array	object
Documents and reports that are part of the implementation phase e.g. audit and evaluation reports. See Document		

Identifier

The identifier block provides a way to identify the legal entities involved in a contracting process.

Looking to find an organisation identifier? Go to http://org-id.guide

scheme	string,null
Organization identifiers should be drawn from an existing organization identifier list. The scheme field is used to indicate the list or register from which the identifier is drawn. This value should be drawn from the Organization Identifier Scheme codelist.	
id	string,null
The identifier of the organization in the selected scheme.	
legalName	string,null
The legally registered name of the organization.	
uri	string(uri),null
A URI to identify the organization	

Item

id	string
A local identifier to reference and merge the items by. Must be unique within a given array of items.	
internalId	string
The identifier used for sequence of items	
description	string,null
A description of the goods, services to be provided.	

classification		object
The primary classification for the item. See the [itemClassificationScheme] See Classification		
additionalClassifications	array	object
An array of additional classifications for the item. See the [itemClassificationScheme] See Classification		
quantity		number,null
The number of units required		
unit		object
A description of the unit in which the supplies, services or works are provided (e.g. hours, kilograms) and the unit-price. See Unit		
relatedLot		string,null
If this item belongs to a lot, provide the identifier of the related lot here.		
deliveryAddress		object
See Address		

JointProcurement

isJointProcurement	boolean
A True/False field to indicate if this is a joint procurement or not. Required by the EU	
country	string,null
ISO Country Code of the country where the law applies. Use ISO Alpha-2 country codes.	

LegalProcedure

id	string,null
Legal identifier(s) of any review procedure(s) initiated. Required by the EU	
title	string,null
Title(s) of any legal proceedings(s) initiated.	
uri	string(url),null
Legal Proceedings uri	

Lot

id	string
A local identifier for this lot, such as a lot number.	
internalId	string
The identifier used for sequence of lots	
title	string
A title for this lot.	
description	string
A description of this lot.	
status	
The current status of the process related to this lot based on the tenderStatus codelist	
statusDeails	string
The current detailed status of the process related to this lot based on the tenderStatusDetails codelist	
value	float
The maximum estimated value of this lot.	

See Value

hasOptions		boolean
A True/False field to indicate if lot options will be accepted		
hasVariants		boolean
A True/False field to indicate if lot variants will be accepted.		
hasRenewals		boolean
A True/False field to indicate whether contract renewals are allowed.		
recurrentProcurement.isRecurrent		boolean
A True/False field to indicate whether this is a recurrent procurement		
contractPeriod		object
The period over which the contract is estimated or required to be active for this lot See Period		
placeOfPerformance		object
The address and further description of the place where the contract will be performed See PlaceOfPerformance		

Metric

id		string
An identifier for this metric		
title		string
The title of this metric		
description		string
A short description of the metric. This may include short details of measurement methods		
observations	array	object
An array of target or actual values for this metric See Observation		

Milestone

id		string
A local identifier for this milestone, unique within this block.		
title		string
Milestone title		
type		string
The type of milestone, drawn from an extended milestoneType codelist		
description		string
A description of the milestone		
code		string
Milestone codes can be used to track specific events that take place for a particular kind of contracting process.		
dueDate		string, date-time
The date the milestone is due		
dateMet		string, date-time
The date on which the milestone was met		
dateModified		string, date-time
A description of the milestone		
status		string
The status that was realized on the date provided in dateModified, drawn from the milestoneStatus codelist		

documents	array	object
List of documents associated with this milestone See Document		
relatedLots	array	string
If this milestone relates to a particular lot, provide the identifier(s) of the related lot(s) here.		
relatedParties	array	object
Parties that have a relationship with the milestone See OrganizationReference		
additionalInformation		string
Additional information about the milestone		

Observation

id		string
A local identifier for this specific observation.		
period		object
The period over which this observation is measured See Period		
value		object
For financial metrics, the value of this forecast, target or actual observation see Value		
measure		string
For non-financial metrics, the measure of this forecast, target or actual observation. Free text or numerical values.		
unit		object
Unit of measure for this observation See Unit		
notes		string
Any notes on this observation. This may include clarifying information.		

Organization

name		string
Common Entity name		
id		string
The ID used for cross-referencing to this party from other sections of the release.		
identifier		object
The primary identifier for this organization. Identifiers that uniquely pick out a legal entity should be preferred. See Identifier		
additionalIdentifiers	array	object
A list of additional / supplemental identifiers for the organization See Identifier		
address		object
See Address		
contactPoint		object
See contactPoint		
roles	array	string

The party's role(s) in the contracting process. Role(s) should be taken from the [partyRole codelist]		
details		string, date-time
Additional classification information about parties. See Details		
buyerProfile		string, uri
For buyer organisations only: the url of the organization's buyer profile. Specified by the EU		
persones	array, null	object
Individuals who represent this legal entity in this specific contracting process. See Persone		

OrganizationReference

name		string
Common Entity name		
id		string
The ID used for cross-referencing to this party from other sections of the release.		
identifier		object
The primary identifier for this organization. Identifiers that uniquely pick out a legal entity should be preferred. See Identifier		
additionalIdentifiers	array	object
A list of additional / supplemental identifiers for the organization See Identifier		
address		object
See Address		
contactPoint		object
See contactPoint		

Period

startDate		string, date-time
The start date for the period		
endDate		string, date-time
The end date for the period. Has to be later than ‘startDate’ (at least one day)		
durationInDays		integer
The maximum duration of this period in days.		

Permit

scheme		string
Identifier that uniquely pick out a legal registry consists needed details. By default here - MD-SRL		
id		string
Identifier of permit/license in selected scheme		
description		string

The human readable name of permit/license in selected scheme		
permitDetails Details about permit/license See PermitDetails		object
uri A URI to identify the permit/license		string

PermitDetails

issuedBy Entity who issued this permit/license See OrganizationReference		object
validityPeriod Period of validity of this permit/license. See Period		string
issuedThought Reference to an activity (described in 'mainEconomicActivities') which is covered by this permit/license See EconomicActivityReference		string

Persone

title Title of the person (e.g. Mr, Mrs. Ms)		string
name Full name of the person		string
identifier Person’s identifier See Identifier		object
businessFunction The list of functions of this person for specific organization See BusinessFunction	array	object

PersoneReference

id Person’s identifier		string
name Full name of the person		string

PlaceOfPerformance

address See Address	object
description Further description of the place of performance of the contract.	string

Planning

budget		object
Information about the budget line, and associated projects, through which this procurement is funded. See Budget		
rationale		string
The rationale for the procurement provided in free text. More details can be provided in an attached document		
documents		string
A list of documents related to the planning process. See Document		
implementation		object
See Implementation		
milestones	array	object
Any milestones related to this planning See Milestone		

Unit

name		string
Name of the unit		
id		string
The identifier from the codelist referenced in the scheme property		
scheme		string
The list from which units of measure identifiers are taken.		
value		object
The monetary value of a single unit. See Value		

RelatedProcess

identifier	string
The OCID of the related process	
uri	string, uri
A URI pointing to an OCDS release or record for the related process. This should be a machine-readable release or record package containing the relevant ocid	
id	string
A local identifier for this relationship, unique within this array	
relationship	string
Specify the type of relationship using the [related process codelist]	
title	string
The title of the related process, where possible this field should match the tender/title field in the related process	
scheme	string
The identification scheme used by this cross-reference - “ocid”	

Requirement

id	string
The identifier for this requirement	
title	string
Requirement title	

description		string
Requirement description		
dataType		string
"string", "date-time", "number", "integer		
expectedValue		string, boolean, number, integer
Requirement description		
minValue		according to dataType
Used to state the lower bound of the requirement when the response must be within a certain range		
maxValue		according to dataType
Used to state the upper bound of the requirement when the response must be within a certain range		
period		according to dataType
A requirement response provides the value for the requirement and may provide the period to which it applies See Period		

RequirementGroup

id		string
The identifier for this requirement group		
description		string
Requirement group description.		
requirements	array	object
A list of requirements which must all be satisfied for the requirement group to be met See Requirement		

RequirementResponse

id		string
The identifier for this requirement response		
title		string
Requirement response title		
description		string, null
Requirement response description		
value		according to requirement.dataType
Requirement response value. The value must be of the type defined in the requirement.dataType field		
requirement		object
The id and title of the requirement which the response is applicable to See RequirementReference		
relatedTenderer		object
Reference the entry in the parties section for the tenderer the response relates to See OrganizationReference		
period		object, null
The period which the requirement response is applicable to See Period		
responder		object
The object for the specific person from the responding Organization See Persone		

RequirementReference

id		string

The id of the requirement which the response is applicable to		
title		string
The title of the requirement which the response is applicable to		

RequestGroup

id		string
The identifier for this request group		
description		string
Request group description.		
requirements	array	object
A list requests which must all be satisfied for the requirement group to be met See Request		

Request

id		string
The identifier for this request		
title		string
Request title		
description		string
Request description		

Transaction

id		string
A unique identifier for this transaction.		
source		string(url), null
Used to point either to a corresponding Fiscal Data Package or machine-readable source		
date		string(date-time), null
The date of the transaction		
value		object
The value of the transaction. See Value		
executionPeriod		object
Period during which this transaction should be executed then related contract milestone is achieved See Period		
uri		string(url), null
A URI pointing directly to a machine-readable record about this spending transaction.		
payer		object
The organization from which the funds in this transaction originate See OrganizationReference		
payee		object
The organization which receives the funds in this transaction See OrganizationReference		
budgetSource	array	object
See BudgetSource		
type		string
The type of payment, specified with one of the values from transactopnTypes codelist		
relatedContractMilestone		string

The identifier of the milestone of contract on which this transaction is relates on

ReviewProceedings

buyerProcedureReview		boolean
A True/False field to indicate if an economic operator applied to the buyer for a review of the procedure.		
reviewBodyChallenge		boolean
True/False to indicate if an economic operator or another party challenged the procedure before a review body.		
legalProcedures	array	object
Identifier(s) of any review procedure(s) initiated. Required by the EU See LegalProcedure		

Value

amount		number
Amount as a number. It should be considered that this field describes amount excluding VAT everywhere except contracts. For contracts this field describes total amount (including all taxes)		
currency	enum	string
The currency for each amount should always be specified using the uppercase 3-letter currency code from ISO4217.		
valueAddedTaxIncluded		boolean
True/False value indicates whether this value includes taxes or not. Applicable on the contracting stage		
amountNet		number
Amount excluding VAT. Applicable on the contracting stage		

Verification

type	enum	string
value		string

4.1.6 Codelists

For more information about entitities definitions, you can find the extended version of this section in the latest version of the document *Codelists.xlsx*.
Copied from eProcurementSystem.MD:

Release Tag

A contracting process may result in a number of releases of information over time. These should be tagged to indicate the stage of the contracting process they relate to.

planning	A contracting process is proposed and/or in the planning stage. Information in the tender section describes the proposed process and description of subject..
planningUpdate	Details of a proposed contracting process are being updated.
tender	Providing information about a new tender (call for proposals) process.
tenderAmendment	An amendment to an existing tender release.

tenderUpdate	An update to an existing tender release.
tenderCancellation	The cancellation of an existing tender.
award	Providing information about the award of a contract.
awardUpdate	An update to an existing award release.
awardCancellation	Providing information about the cancellation of an award.
contract	Providing information about a contract that has been/will be entered into.
contractUpdate	Providing information about updates to a contract.
contractAmendment	An amendment to an existing contract release.
implementation	Providing new information on the implementation of a contracting process.
implementationUpdate	Updating information provided about the implementation of a contract
contractTermination	Providing information at the end of a contracting process.
compiled	Used in compiled records, which have merged together multiple releases to provide a snapshot view of the contract, and a version history.

Initiation Type

Contracting processes may be formed under a number of different processes. Currently, only 'tender' is supported in this codelist. Future versions of the standard may support other Initiation Types.

tender	An open competitive bidding or tendering to form contracts.

Tender Status

The tender.status field is used to indicate the current status of a tender process.

planning	A future contracting process is being considered
planned	A contracting process is scheduled, but is not yet taking place
active	A tender process is currently taking place
cancelled	The tender process has been cancelled.
unsuccessful	The tender process was unsuccessful.
complete	The tender process is complete.
withdrawn	No further information on this process is available under this ocid.

Tender Status Details

The tender.statusDetails field is used to indicate the current state of a tender process.

active.preselection
active.preselected
active.pre-qualification
active.pre-qualified
active.clarification
active.tendering
active.evaluation
active.evaluated
active.execution
active.suspended
complete.empty

Fundings Statuses and its details

Expenditure Item Status

The tender.status field is used for EIs to indicate the current status of specific item

planning
active
compete
cancelled

Expenditure Item Status Details

The tender.statusDetails field is used for EIs t to indicate the current state of specific item.

blocked

Funding Source Status

The tender.status field is used for FSs to indicate the current status of specific item

planning	source announced for future use
planned	source announced and sent for external approve (treasury)
active	source announced and in use
unsuccessful	source received from validator (treasury) with negative answer
complete	source is spent or budget period expired
cancelled	source cancelled

Funding Source Status Details

The tender.statusDetails field is used for FSs t to indicate the current state of specific item.

blocked	used together with ‘active’ status: budget period expired but there are still contracting process in ‘active’ status which use funding from this source

Procurement Methods

The procurement method is the procedure used to purchase the relevant works, goods or services.

open	All interested suppliers may submit a tender.

selective	Only qualified suppliers are invited to submit a tender.
limited	The procuring entity contacts a number of suppliers of its choice.
direct	The contract is awarded to a single supplier without competition.

Procurement Methods Details

Additional detail on the procurement method used.

microValue	Micro Value Tender
openTender	Open Tender
restrictedTender	Restricted Tender
frameworkAgreement	Framework Agreement
smallValue	Small Value Tender
directAward	Direct Award Contract
negotiation	Negotiation Procedure

Procurement Category

The codelist is used to indicate the primary focus of a contracting process.

goods	The primary subject of this procurement involves goods or supplies.
works	The primary subject of this procurement involves construction, repair, rehabilitation, demolition, restoration or maintenance of some asset
services	The primary subject of this procurement involves professional services of some form, generally contracted for on the basis of measurable outputs

Extended Procurement Category

The codelist is used to provide additional detail about the focus of a contracting process.

goods	This procurement involves physical or electronic goods or supplies
works	This procurement involves construction, repair, rehabilitation, demolition, restoration or maintenance of some asset or infrastructure.
services	This procurement involves professional services of some form, generally contracted for on the basis of measurable outputs or deliverables.
consultingServices	This procurement involves services provided on a consultancy basis.

Award Criteria

The codelist describes the basis on which contract awards will be made.

priceOnly	The award will be made to the qualified bid with the lowest price.
costOnly	The award will be made to the qualified bid with the lowest overall cost.
qualityOnly	The award will be made to the qualified bid with the highest quality against some assessment method.
ratedCriteria	The award will be made to the qualified bid with the highest score against a set of weighted criteria such as price, cost and quality.

Award Criteria Details

The codelist for award criteria processing format

manual	the awarding will be approached manually
automated	the awarding will be approached automatically based on ‘awardCriteria’ and set of relevant requirementResponses received from the tenderers against `requirements` applied by PE

Submission Method

The codelist is used to identify the mechanism through which a submission may be made.

electronicSubmission	Bids will be received through an electronic procurement platform.
electronicAuction	Bids will be received through an electronic auction platform.
written	Bids will be received via written documents, delivered as hard copies
inPerson	Bids will only be received if they are hand delivered.

Related Process

The related process block is used at the release level to point backwards to prior processes and at the contract level to point onward to sub-contracts, renewal or replacement processes. Codelist determines the kind of relationship that is being described.

framework	Framework agreement procedure first stage.
planning	This contracting process follows on from the related planning process.
parent	This contracting process may result in a sub-contract of the related process.
prior	This contracting process is the renewal of the related prior process.
unsuccessfulProcess	This contracting process follows on from a previous unsuccessful process.
subContract	The related process may result in a sub-contract of this contract.
replacementProcess	The related process may result in the replacement of this contract.
renewalProcess	The related process may result in the renewal of this contract.
x_fundingSource	
x_expenditureItem	
x_planned	
x_preselection	
x_prequalification	
x_evaluation	
x_contracting	
x_directContracting	
x_execution	

Legal Basis

DIRECTIVE_2014_23_EU
DIRECTIVE_2014_24_EU
DIRECTIVE_2014_25_EU
DIRECTIVE_2009_81_EC
REGULATION_966_2012
NATIONAL_PROCUREMENT_LAW

Submission Method Rationale

TOOLS_DEVICES_FILE_FORMATS_UNAVAILABLE
IPR_ISSUES
REQUIRES_SPECIALISED_EQUIPMENT

PHYSICAL_MODEL
SENSITIVE_INFORMATION

Submission Languages

en
ro
ru

Award Status

An award move through multiple statuses. Releases over time may update the status of an ‘award’.

pending	This award has been proposed, but is not yet in force.
active	Bid under this award has been selected, and decision in force
cancelled	This award has been cancelled.
unsuccessful	Bid under this award has been disqualified and decision in force

Country

ISO-ALPHA2	Country Codes - ISO 3166

Region

ISO-ALPHA2	Subdivisions Codes - ISO 3166
MD-CUATM	Classifier of administrative-territorial units (MD)

Locality

MD-CUATM	Classifier of administrative-territorial units (MD)
OTHER	Free definition In case UN/LOCODE is not applicable

Banks schemes

MD-BNM	State register of licensed banks (MD)

Bank account identification schemes

MD-IBAN	International Bank Account Number (MD)

Award Status Details

An award move through multiple states (disclosed as ‘statusDetails’). Releases over time may update the ‘statusDetails’ of an award.

active	award will go to ‘active’ status
cancelled	
unsuccessful	award will go to ‘unsuccessful’ status
consideration	award is under consideration

empty	

Bid Status

Bid move through multiple statuses. Releases over time may update the ‘status’ of an ‘bid’.

invited	A bid has been invited from the listed tenderer(s).
pending	A bid has been submitted but not yet evaluated.
valid	The submitted bid met the qualification requirements.
disqualified	The submitted bid did not meet the qualification requirements
withdrawn	The submitted bid was withdrawn by the tenderer or because of terms.

Bid Status Details

valid	
disqualified	

Bid Statistics

Bids

requests	The total number of unique requests to participate received
bids	The total number of unique bids received.
validBids	The total number of unique bids received that were considered valid against.

Bidders

bidders	The total number of unique organizations submitting bids
qualifiedBidders	The total number of unique organizations passing the qualification stage
disqualifiedBidders	The total number of unique organizations that did not pass the qualification stage

EU

electronicBids	The number of bids received by electronic means.
smeBids	The number of bids received from Small and Medium Sized Enterprises
foreignBids	The number of bids received from bidders from outside the country
foreignBidsFromEU	The number of bids received from bidders from outside the country, but based in EU
tendersAbnormallyLow	The number of tenders excluded because they were abnormally low.

Item Classification Scheme

CPV	General codes of Common Procurement Vocabulary
CPVS	Supplementary codes of Common Procurement Vocabulary

Criterion and Request Sources

tenderer	Criterion on request came from the tenderer
buyer	Criterion on request came from the buyer
procuringEntity	Criterion on request came from the Procuring Entity

Criterion Relations

item	

tenderer	
lot	

Confirmation Request Relations

document	This confirmation request is related to specific document
amendment	This confirmation request is related to specific amendment

Contract Award Notice Statuses

pending	This CAN has been issued and related awarded contract is expected
active	This CAN has been issued and related awarded contract is concluded
cancelled	This CAN has been cancelled before conclusion of related awarded contract
unsuccessful	

Contract Award Notice StatusDetails

contractProject	The preparation of related awarded contract is in progress

Contract Statuses

pending	This contract has been proposed, but is not yet in force.
active	This contract has been signed by all the parties, and is now legally in force.
cancelled	This contract has been cancelled prior to being signed.
terminated	This contract was signed and in force, and has now come to a close.

Contract Status Details

contractProject	This contract has been issued as a draft and needs to be prepared by CA
issued	This contract has been fully prepared by CA and ready for approvement by CA
approvement	This contract is under approvement by CA
approved	This contract has been signed by CA and moved for signing by EO
signed	This contract has been signed by both, CA and EO
verification	This contract has been sent for verification to the State Treasury
verified	This contract has been verified by State Treasury
clarification	This contract has been returned by State Treasury with clarification request
rejected	This contract has been rejected by State Treasury
cancellation	This contract has been set for cancellation (termination before signed by Supplier)
termination	This contract has been set for termination
complete	This contract has been terminated as completed
unsuccessful	This contract has been terminated as unsuccessful

Conversion Relations

requirement	Where conversion is related to the specific requirement

Document Type

Planning

procurementPlan	Documentation that sets out the basis for this particular contracting process.
riskProvisions	Documents covering how risks will be managed as part of this process.

Tender

tenderNotice	The formal notice that gives details of a tender.
biddingDocuments	Documentation for potential suppliers, describing the goals of the contract
technicalSpecifications	Detailed technical information about goods or services to be provided.
evaluationCriteria	Documentation on how bids will be evaluated.
evaluationReports	Documents on the evaluation of the bids and the application of the evaluation criteria,
contractDraft	A draft or pro-forma copy of the contract.
eligibilityCriteria	Detailed documents about the eligibility of bidders.
clarifications	Documentation that provides replies to issues raised.
shortlistedFirms	Documentation providing information on shortlisted firms.
contractArrangements	Documentation of the arrangements for ending the contract(s).
complaints	Documentation of any complaints received, or decisions in response.
billOfQuantity	Itemized information on materials, parts and labour, terms and conditions for provision.
contractGuarantees	Documentation of guarantees relating to a contracting process or contract.
conflictOfInterest	Documentation of conflicts of interest declared or uncovered.
illustration	Images intended to provide supporting information.
cancellationDetails	Documentation of reasons for cancellation of a process, award or contract.

Bid

illustration	Images intended to provide supporting information.
submissionDocuments	Documentation submitted by a bidder as part of their proposal.
x_commercialOffer	Commercial part of the offer
x_qualificationDocuments	Qualification documents
x_eligibilityDocuments	Eligibility documents
x_technicalDocuments	Technical document of the offer

Award

awardNotice	The formal notice that gives details of the contract award.
contractDraft	A draft or pro-forma copy of the contract.
contractArrangements	Documentation of the arrangements for ending the contract(s).
contractSchedule	Any document which contains additional terms.
winningBid	Documents of the winning bid
complaints	Documents of any complaints received, or decisions in response to complaints.
conflictOfInterest	Documents of conflicts of interest declared or uncovered.
cancellationDetails	Documents of reasons for cancellation of a process, award or specific contract.

Contract

contractNotice	The formal notice that gives details of a contract being signed and valid to start implementation.
contractSigned	A copy of the signed contract. Consider providing both machine-readable and a separate document entry
contractArrangements	Documentation of the arrangements for ending the contract(s).
contractSchedule	Any document which contains additional terms or
contractAnnexe	Copies of annexes and other supporting documentation related to the contract.
contractGuarantees	Documentation of guarantees relating to a contracting process or contract.
subContract	Documentation detailing subcontracts and/or providing a copy of subcontracts
contractSummary	Documentation providing an overview of the key terms and sections of the contract. Commonly used for
conflictOfInterest	Documentation of conflicts of interest declared or uncovered.
illustration	Images intended to provide supporting information.
cancellationDetails	Documentation the arrangements, or reasons, for cancellation of a contract

Implementation

conflictOfInterest	Documentation of conflicts of interest declared or uncovered.
illustration	Images intended to provide supporting information.
cancellationDetails	Documents of reasons for cancellation of a process, award or contract.

Business Function

regulatoryDocument	Documents which confirm specific business function or activity

Form Service Parameters

The following list describes Form Service params used

'lang' codes

EN	English
RU	Russian
RO	Romanian

'country' codes

MD	Republic of Moldova

'form' codes

EI	Expenditure Item
UPDATE-EI	Update of existing EI
FS	Funding Source
UPDATE-FS	Update of existing FS
PN	Planning Notice
UPDATE-PN	Update of existing PN
PIN	Prior Information Notice
UPDATE-PIN	Update of existing PIN
CN	Contract Notice

UPDATE-CN	Update of existing CN
BID	Bid
UPDATE-BID	Update of existing bid
ENQUIRY	Enquiry
ANSWER	Answer for specific enquiry
AWARD	Award
UPDATE-AC	Update of automatically prepared draft of future contract
CANCELLATION-TENDER	Cancellation of contracting process
CANCELLATION-LOT	Cancellation of specific lot of contracting process

'funder' codelist

BUYER
STATE
DONOR

'payer' codelist

BUYER
THIRDPARTY
FUNDER

'procuringEntity' codelist

BUYER
THIRDPARTY

'responsibleContactPerson' codelist

BUYER
THIRDPARTY

'pmd' codelist

MV	Micro Value Procedure
SV	Small Value Procedure
OT	Open Tender Procedure
DA	Direct Award
OP	Other Procedure
NP	Negotiation Procedure

Milestone Type

preProcurement	For events during the planning or pre-procurement phase of a process
approval	For events such as the sign-off of a contract or project.
engagement	For engagement milestones, such as a public hearing.

assessment	For assessment and adjudication milestones
delivery	For delivery milestones
reporting	For reporting milestones, such as when key reports should be provided.
financing	For events such as planned payments, or equity transfers

Milestone Status

scheduled	This milestone has been scheduled
met	This milestone has been achieved.
notMet	This milestone had not been met at the date it was last reviewed
partiallyMet	This milestone was judged to have been partially met at the date it was last reviewed

typeOfBuyer

BODY_PUBLIC	Body governed by public law
EU_INSTITUTION	European institution/agency or international organisation
MINISTRY	Ministry or any other national or federal authority, including their regional or local subdivisions
NATIONAL_AGENCY	National or federal agency/office
REGIONAL_AGENCY	Regional or local agency/office
REGIONAL_AUTHORITY	Regional or local authority

mainGeneralActivity

DEFENCE
ECONOMIC_AND_FINANCIAL_AFFAIRS
EDUCATION
ENVIRONMENT
GENERAL_PUBLIC_SERVICES
HEALTH
HOUSING_AND_COMMUNITY_AMENITIES
PUBLIC_ORDER_AND_SAFETY
RECREATION_CULTURE_AND_RELIGION
SOCIAL_PROTECTION

mainSectoralActivity

AIRPORT_RELATED_ACTIVITIES
ELECTRICITY
EXPLORATION_EXTRACTION_COAL_OTHER_SOLID_FUEL
EXPLORATION_EXTRACTION_GAS_OIL
PORT_RELATED_ACTIVITIES
POSTAL_SERVICES
PRODUCTION_TRANSPORT_DISTRIBUTION_GAS_HEAT
RAILWAY_SERVICES
URBAN_RAILWAY_TRAMWAY_TROLLEYBUS_BUS_SERVICES
WATER

Method modality

electronicAuction	The procurement method involves electronic auction

Party Role

buyer	entity whose budget will be used to pay for goods, works or services related to a contract
procuringEntity	entity managing the procurement, which may be different from the buyer
supplier	The entity awarded or contracted to provide supplies, works or services.
tenderer	All entities who submit a tender
funder	The funder is an entity providing money or finance for this contracting process.
enquirer	A party who has made an enquiry during the enquiry phase of a contracting process.
payer	A party making a payment from a transaction
payee	A party in receipt of a payment from a transaction
reviewBody	A party responsible for the review of this procurement process.

Scale

micro	Micro business
sme	Small or Medium Enterprise
large	Large Enterprise

Auction moves

met	Tenderer made a step in the round within the allotted time
notMet	Tenderer did not take a step in the round and the value of his bid was set automatically

Type Of Supplier

company	
individual	

Transactions Types

advance	Advance payment
payment	Regular payment

Requests Types

digitalSignature	The digital signature is expected as a confirmation
outsideAction	The confirmation of execution of external action is expected

Business Functions

authority	
procurementOfficer	
contactPoint	
technicalEvaluator	responsible for evaluating of the technical part of the offer

technicalOpener	approves the opening of technical part of the offer
priceOpener	approves opening of financial part of the offer
priceEvaluator	

e-GPA

annex1	Central Government Entities
annex2	Sub-central Government Entities
annex3	Other entities (Utilities)

GPA Types of Organizations

subannex101	Ministries/agencies and other government offices
subannex102	Government-owned entity or other national authority
subannex103	Body governed by public law
subannex201	Regional or local administration
subannex202	Regional or local owned entity
subannex301	Utility sector - (A) Production transmission supply storage of gas
subannex302	Utility sector - (B) Production transmission and supply of heat
subannex303	Utility sector - (C) Production transmission and supply of electricity
subannex304	Utility sector - (D) Production transportation and supply of water
subannex305	Utility sector - (E) Functioning of the centralized sewage facilities
subannex306	Utility sector - (F) Provision of services for usage of rail transport infrastructure of general use
subannex307	Utility sector - (G) Functioning of municipal electric transport and operation of its facilities to provide transportation services
subannex308	Utility sector - (H) Provision of services by bus terminals ports and airports
subannex309	Utility sector - (I) Provision of air navigation services for support of movement of airplanes
subannex310	Utility sector - (J) Provision of postal services
subannex311	Utility sector - (K) Geological exploration of the oil gas coal and other types of solid fuel
subannex312	Utility sector - (L) Operation and management of the fixed telecommunication networks of general use or provisio telecommunication services of general use
subannex313	Utility sector - (M) Transportation storage and processing of crude oil and oil products
subannex314	Other - State or community non-commercial (non-profit) organizations
	Other - Commercial organizations with over 50 per cent of government or community shareholding

Amendment statuses

pending
active
cancelled

Amendment Types

cancellation
change

4.1.7 Classifiers

Common Procurement Vocabulary

The main vocabulary is based on a tree structure comprising codes of up to 9 digits associated with a wording that describes the type of supplies, works or services forming the subject of the contract.

scheme	CPV
uri	http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2008:074:0001:0375:EN:PDF

Common Procurement Vocabulary (supplementary)

The supplementary vocabulary may be used to expand the description of the subject of a contract.

scheme	CPVS
uri	https://simap.ted.europa.eu/documents/10184/36234/cpv_sv_2008_explanatory_notes_en.pdf

Classifier of administrative-territorial units (MD)

Classifier is intended for use in the automated processing of information in various fields of activity in the context of administrative-territorial units

scheme	MD-CUATM
uri	http://www.egov

Classifier is available in MDM 

Classifier of organizational and legal forms of economic entities (MD)

This classifier is intended for use in the automated processing of information in various fields of activity in the context of enterprises and organizations on the organizational and legal forms of business entities.

scheme	MD-CFOJ
uri	http://www.egov

Classifier of the types of economic activity (MD)

The classifier is the basis for collecting and providing data on the types of economic activity.

scheme	MD-CTEA
uri	http://www.egov

Classifier of the units of measure (MD)

scheme	MD-CUMA
uri	

Classifier is available in MDM 

4.1.8 International standards

Country Codes - ISO 3166

International Standard for country codes and codes for their subdivisions. The purpose of ISO 3166 is to define internationally recognised codes of letters that we can use when we refer to countries and subdivisions

scheme	ISO-ALPHA2
uri	https://www.iso.org/iso-3166-country-codes.html

United Nations Code for Trade and Transport Locations (UN/LOCODE)

UN Code for Trade and Transport Locations, is a geographic coding scheme developed and maintained by United Nations Economic Commission for Europe. Standard assigns codes to locations used in trade and transport with functions such as seaports, rail and road terminals, airports, etc.

scheme	UN/LOCODE
uri	https://www.unece.org/cefact/locode/service/location

International Standard Classification of Occupations

ISCO-08 provides a system for classifying and aggregating occupational information obtained by means of statistical censuses and surveys, as well as from administrative records.

scheme	ISCO-08
uri	http://www.ilo.org/public/english/bureau/stat/isco/isco08/index.htm

Currency Codes - ISO 4217

The purpose of ISO 4217 is to establish internationally recognised codes for the representation of currencies.

scheme	ISO-4217
uri	https://www.iso.org/iso-4217-currency-codes.html

Languages Codes - ISO 639

The purpose of ISO 639 is to establish internationally recognised codes for the representation of languages.

`scheme`	ISO-639
`uri`	https://www.iso.org/iso-639-language-codes.html

4.1.9 Registers

In order to guarantee a successful implementation of the OCDS eprocurement system, the system needs to be integrated or reuse data from the following state registers (not exhaustive list):

State identification numbers of enterprises and organizations (MD)

scheme	MD-IDNO
uri	http://e-services.md/?q=ru/verifica-idno

Verification service of the IDNO codes is available in M-Connect Daemon

State population register (MD)

scheme	MD-IDNP
uri	http://e-services.md/?q=ru/verifica-idnp

Verification service of the IDNP codes is available in M-Connect Daemon

State register of licences (MD)

scheme	MD-SRL
uri	

Register will soon be available in M-Connect Daemon

State register of legal entities (MD)

scheme	MD-SRLE
uri	

Register will soon be available in M-Connect Daemon

International Bank Account Number (MD)

scheme	MD-IBAN
uri	

State register of licensed banks (MD)

scheme	MD-BNM
uri	https://bnm.md/en/content/authorized-banks-republic-moldova

Register will soon be available in MDM

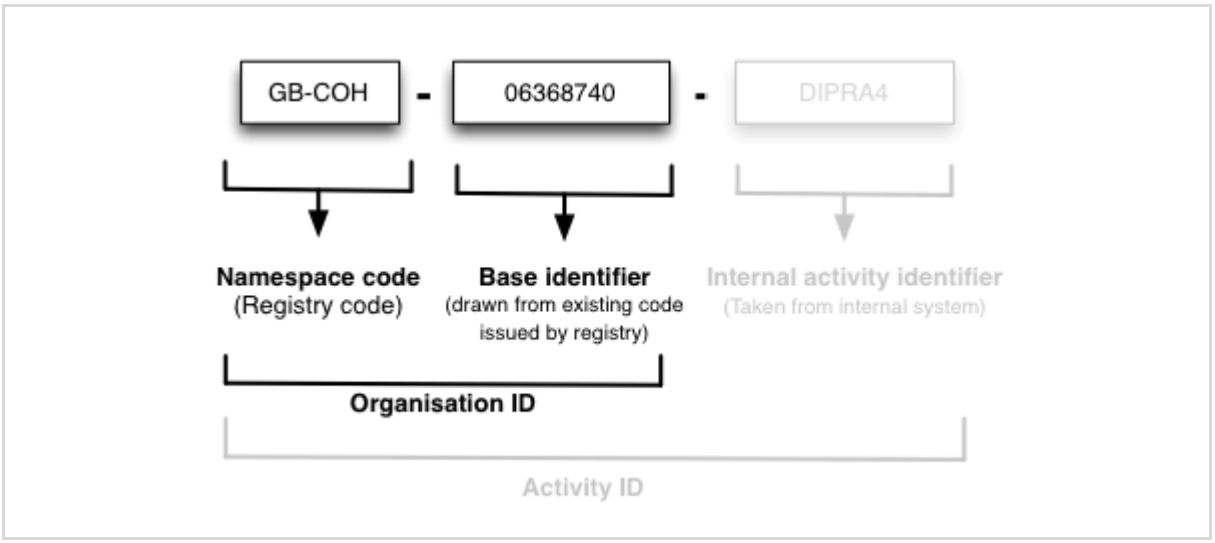
Organization registration registry

The values from this codelist are used to identify the particular list that an organisation identifier was drawn from. The codelist provides a register of known identifier lists, including national company registers, NGO directories and international and multilateral organisation lists - along with guidance and online resources to help locate the identifiers assigned to a specific organisation.

Organisation Identifiers

The IATI standard allows for the recording of information on all organisations that participate in any part of the lifecycle of an aid activity: inter alia donors, beneficiaries, extending and implementing agencies. Crucial to this, is a common process through which to identify and declare identifiers for these organisations.

Two components of IATI Organisation Identifiers



Identifier strings consist of two components:

1. The namespace code - a code that creates and maintains a given organisational identifier registry
2. The base identifier – re-use of an existing identifier from that namespace/registry.

4.2 Procurement Process Stages

Depending on the type of procedure different stages can be used. In general, there are five main stages of the procurement process, from which the competitive phase of the purchasing process can consist (in different combinations and / or sequences):

4.2.1 Groups of the general data-models

4.2.1.1 Budgeting

This stage determines information about the budget available or associated projects, through which specific need can be funded.

Expenditure Item

'Expenditure Item' is a model that describes a group of goods, services or works that a CA plans to procure over a certain period, as well as the amount allocated by the CA to finance the purchases of subject of this group during the specified period. The group is determined by specifying the parent class of the CPV (the first 3 or 4 digits, depending on local specifics in the requirements for determining the subject of procurement). The 'amount' is the sum of all sources of funding identified for this expenditure item as child elements (see below).

'EI' data-model

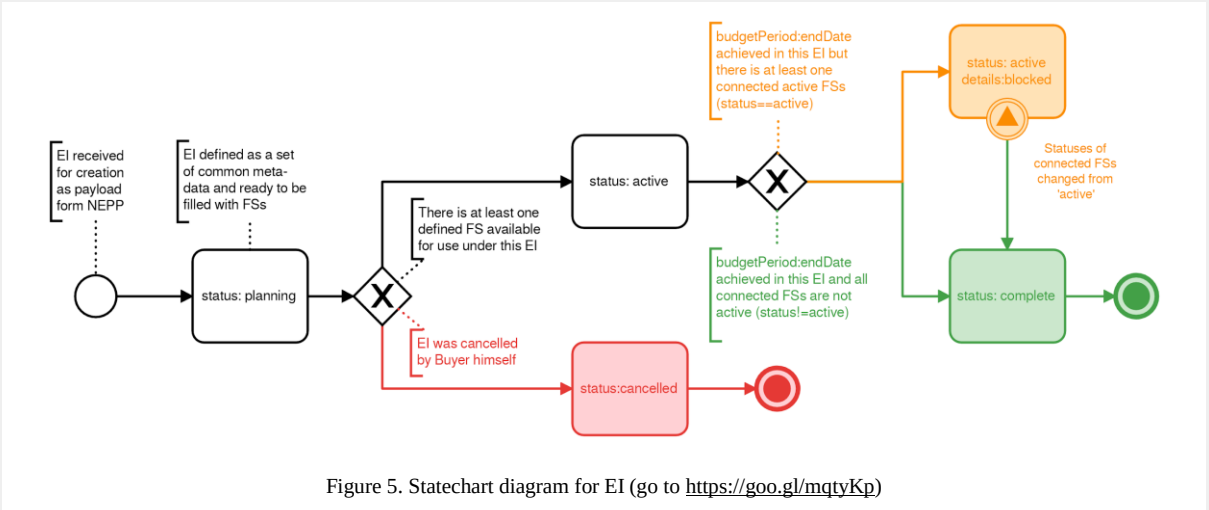


Figure 5. Statechart diagram for EI (go to https://goo.gl/mqtyKp)

Statechart statuses

Expenditure Item statuses

The *status* field is used for EIs to indicate the current status of specific item

Title	Description
`planning`	EI is in planning phase and no active contracting processes related
`active`	EI is under implementation due to at least one active contract process related

`compete`	EI is closed due to fulfillment of needs described
`cancelled`	EI is cancelled Funding Source

Expenditure Item Status Details

The *statusDetails* field is used for EIs t to indicate the current state of specific item.

Title	Description
`blocked`	EI is suspended due to expiration of budget periods of related funding sources

Query-model

'EI' describes as Release Package according to the following scheme:

Title, description	Type
`Ocid` A globally unique identifier for this Expenditure Item	string
`Id` An identifier for this particular release of information for EI	string
`Date` The date this information is released	date-time
`Tag` A value from the releaseTag codelist that identifies the nature of the release being made	string
`initiationType` String specifying the type of initiation process used for this contract	string
`tender.id` An identifier for this Expenditure Item	string
`tender.title` Title for this expenditure item	string
`tender.description` Description for this expenditure item	string
`tender.mainProcurementCatogory` The primary category describing the main object of this tender.	string
`tender.classification` Object described information about Entity in whose interests this EI is initiated See Classification	object
`tender.status` The current status of the EI based on the table of EI statechart statuses	string
`tender.statusDetails` The current status of the EI based on the table of EI statechart statuses	string
`planning.budget.id` An identifier for the definition of subject of procurement under this EI (based on CPV)	string
`planning.budget.period` Object described the period of validity of this EI See Period	object

planning.budget.amount		object
Object described the sum of available funds under this EI See Value		
planning.rationale		string
The rationale for the EI provided in free text.		
relatedProcesses	array	object
Objects described information about related funding (FSs) and contracting processes based on this EI See RelatedProcess		
Parties	array	object
Object described information about Entity in whose interests this EI is initiated See Organization		
Buyer		object
An entity whose interest will be considered to pay for goods, works or services related to this EI See OrganizationReference		

Command model

To create new entity of EI standard model should be used for preparation of POST-request for BPE.

Attribute	Description	Obligation
tender.title	Has to be specified as a free text	Optional
tender.description	Has to be specified as a free text	Optional
tender.classification	Has to be specified as a free text	Mandatory
planning.budget.period	Has to be specified as an object according to definition	Mandatory
planning.rationale	Has to be specified as a free text	Optional
Buyer	Has to be specified as an object according to definition	Mandatory

Update model

There are several options of executing for updating an existing entity of an EI depending on funding and current status of FSs connected to this EI:

- EI created but no relevant FSs yet defined (status:planning) - such EI available for *update*
- EI created and at least one relevant FS defined (status:active) - such EI available for *amendment*

Update

To update of existing entity of EI cutted command model (update model) should be used for preparation of POST-request for BPE

Title	Description	Obligation
tender.title	Can be amended as a free text	Optional
tender.description	Can be amended as a free text	Optional
planning.rationale	Can be amended as a free text	Optional

`tender.classification`	Can be amended until no FS related	Mandatory
`planning.budget.period`	Can be amended	Mandatory

Amend model

To amend existing EI the next model should be used for preparation of POST-request for BPE.

Title	Description	Obligation
`tender.title`	Can be amended as a free text	Optional
`tender.description`	Can be amended as a free text	Optional
`planning.rationale`	Can be amended as a free text	Optional

Funding Source

'Funding Source' is a model that describes a specific source of funding for needs from the parent 'Expenditure Item'. Description contains information on both amount and the organization that provides the funds as well as other parties involved in the disposal of this funds: payer, donor. Also, through this model, the current status of the described budget is displayed, which makes it possible to understand whether the given budget is approved by the providing party (Treasury or Donor) and to what extent. The valence of the model enables the organisation of the relationships of the entities with other model entities (such as tenders and contracts). Thus, there is an opportunity to display the source of funds in announced tenders or concluded contracts.

'FS' data-model

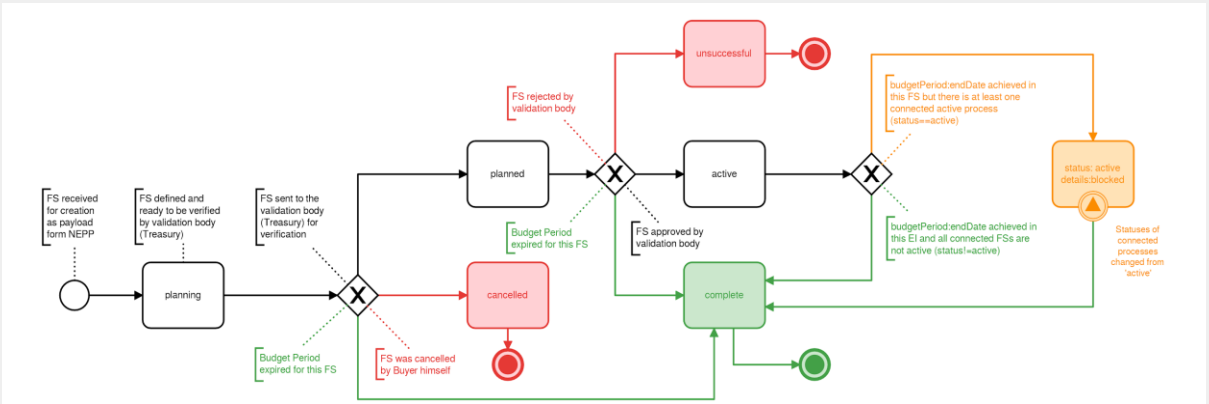


Figure 6. Statechart diagram for FS (go to https://goo.gl/EYDyjT)

Statechart statuses

Funding source statuses

The *status* field is used for FSs to indicate the current status of specific item

Title	Description

`Planning`	source announced for future use
`Planned`	source announced and sent for external approve (treasury)
`Active`	source announced and in use
`Unsuccessful`	source received from validator (treasury) with negative answer
`Complete`	source is spent or budget period expired
`Cancelled`	source cancelled

Funding source status details

The *statusDetails* field is used for FSs t to indicate the current state of specific item.

Title	Description
`Blocked`	used together with 'active' status: budget period expired but there are still contracting process in 'active' status which use funding from this source

Query model

'FS' describes as Release Package according to the following scheme:

Title, description	Type
`ocid` A globally unique identifier for this Funding Source	string
`id` An identifier for this particular release of information for Funding Source	string
`date` The date this information is released	date-time
`tag` A value that identifies the nature of the release being made	string
`initiationType` String specifying the type of initiation process used for this contract	string
`tender.id` An identifier for this FS	string
`tender.status` The current status of the FS based on table of FS statechart statuses	string
`tender.statusDetails` The current status of the FS based on table of FS statechart statuses	string
`planning.budget.id` An identifier for the budget line item which provides funds for this FS.	string
`planning.budget.period` Object described the period of availability of this FS See Period	object
`planning.budget.amount` Object described the sum of available funds under this FS See Value	object
`planning.budget.sourceEntity` Reference the organization providing the budget See OrganizationReference	object
`planning.budget.isEuropeanUnionFunded` A True/False field to indicate whether this procurement is related to a project and/or programme financed by EU	boolean
`planning.budget.project`	string

The name of the project that through which this funds are taken (if applicable)		
planning.budget.projectID		
An external identifier for the project that this funds forms part of, or is funded via (if applicable).		
planning.budget.uri		uri
A URI pointing directly to a machine-readable record about the budget line-item or line-items that fund this FS		
planning.budget.verified		boolean
A True or False field to indicate whether this funds are verified by authorized entity (Treasury)		
planning.budget.verificationDetails		string
The rationale for the verification provided in free text.		
planning.rationale		string
The rationale for the FS provided in free text.		
relatedProcesses	array	object
Objects described information about parent EI and related contracting processes and conducted contracts See RelatedProcess		
parties	array	object
Object describes information about organizations involved in the disposal of this particular budget: payer, donor. See Organization		

Command model

To create new entity of FS standard model should be used for preparation of POST-request for BPE:

Title	Description	Obligation
tender.procuringEntity	Has to be specified as an object according to definition	Mandatory
buyer	Has to be specified as an object according to definition	Mandatory
planning.budget.id	Has to be specified according to budget classification	Mandatory
planning.budget.description	Has to be specified as a free text	Optional
planning.budget.period	Has to be specified as an object according to definition	Mandatory
planning.budget.amount	Has to be specified as a number	Mandatory
planning.budget.isEuropeanUnionFunded	Has to be indicated	Mandatory
planning.budget.europeanUnionFunding	Has to be specified as an object according to definition	Optional
planning.budget.project	Has to be specified as an object according to definition	Optional
planning.budget.projectID	Has to be specified according to budget classification	Optional
planning.budget.uri	Has to be specified according to URI data-format	Optional
planning.rationale	Has to be specified as a free text	Optional

Update model

The update of existing entity of FS has several options of execution depending of source of funding and current status of FS itself. Thus, in case if FS that needs to be updated describes

State funding (allocated by State Treasury), depending of current status from its validation perspective could be changed in a one of next ways:

- FS was recently created and either not send for initial validation yet or still under consideration (status:planning) - such FS available for *update*
- FS successfully passed the validation and available for use (status:active) such FS available for *amendment*

In case if FS that needs to be updated describes own fundings, it is available for direct amending because it is initially in active status.

Update and amend

To update or amend existing FS the next model should be used for preparation of POST-request for BPE

Title	Description	Obligation
`planning.budget.description`	Can be amended as a free text	Optional
`planning.budget.period`	Can be amended as an object according to definition	Mandatory
`planning.budget.amount`	Can be amended specified as a number	Mandatory
`planning.budget.isEuropeanUnionFunded`	Can be amended	Mandatory
`planning.budget.europeanUnionFunding`	Can be amended as an object according to definition	Optional
`planning.budget.project`	Can be amended as an object according to definition	Optional
`planning.budget.projectID`	Can be amended	Optional
`planning.budget.uri`	Can be amended according to URI data-format	Optional
`planning.rationale`	Can be amended as a free text	Optional

Budgeting transport model

On the transport level data-entities describing the budgeting could be organized and transferred (published) as a Record package[2] according to OCDS. Therefore ‘EI‘ can be described as Record Package with linkage to Release Package with separate Release Package describes details of EI and Release Packages of used FSs:

[2] http://standard.open-contracting.org/latest/en/schema/record_package/

```
{
  "uri":                                                                    "",
  "version":                                                                "",
  "extensions":                                                             [],
  "publisher":                                                              {},
  "license":                                                                "",
  "publicationPolicy":                                                      "",
  "publishedDate":                                                          "",
  "packages":                                                               [
    "uri                  of                  EI                  release                  package",
    "uri(s)            of            EI's            FS(s)            release            package(s)"
  ],
  "records":                                                                [
    {
      "description":    "This    Compiled    release    includes    definition    of    EI    details",
      "properties":                                                         {
        "ocid":                                                            "",
        "compiledRelease":                                                 {
          "$ref":                                                        "#/models/EI"
        }
      }
    },
    {
      "description":    "This    compiled    release    includes    definition    of    FS    details",
      "properties":                                                         {
        "ocid":                                                            "",
        "compiledRelease":                                                 {
          "$ref":                                                        "#/models/FS"
        }
      }
    }
  ]
}
```

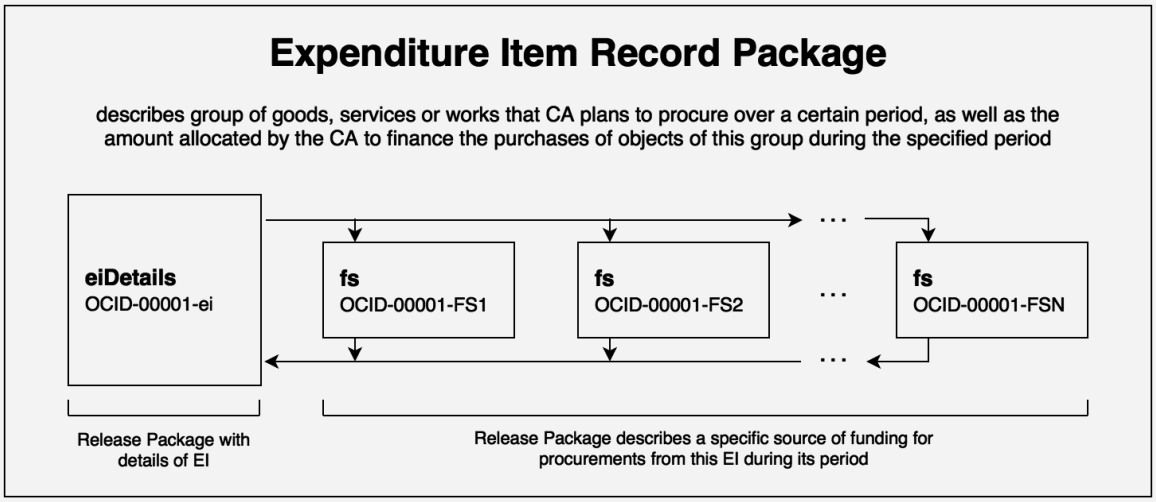


Figure 6.1. Scheme of a budget description using EI and FS models

4.2.1.2 'Multi-stage process' concept

Multi-stage process (MSP) is a model that describes a specific contract process - the procurement procedure announced by the CA. Depending on the type of procedure, method of

procurement, geography and the legal basis, the attribute composition of the model can be adjusted, but its general logic remains unchanged for all types of procedures.

MSP described with a Record Package consists of four (at least) or more Release Packages:

1. Contracting Process (CP); overarching hi-level common data and the budget of procurement
2. Release package with all other data. Depends of start point: PN, PIN or CfC (PS/PQ/EV)
3. Expenditure Item (EI) - hi-level common data of used EI
4. At least one Funding Source (FS) - detailed information about used FS

Transport model

```
{
  "uri":                                                                    "",
  "version":                                                                "",
  "extensions":                                                             [],
  "publisher":                                                              {},
  "license":                                                                "",
  "publishedDate":                                                          "",
  "packages":                                                               [
    "uri                  of                  EI                  release                  package",
    "uri(s)                of                 FS(s)                release                package(s)",
    "uri             of             CP             hi-level             release             package",
    "uri             of             CN             details              release             package"
  ],
  "records":                                                                [
    {
      "description":       "This       Compiled       release       includes       CN's       hi-level",
      "properties":                                                         {
        "ocid":                                                            "",
        "compiledRelease":                          {"$ref":                          "#/models/CP"}
      }
    },
    {
      "description":       "This       compiled       release       includes       CN's       details",
      "properties":                                                         {
        "ocid":                                                            "",
        "compiledRelease":                          {"$ref":                          "#/models/CN"}
      }
    },
    {
      "description":       "This       compiled       release       includes       FS       details",
      "properties":                                                         {
        "ocid":                                                            "",
        "compiledRelease":                          {"$ref":                          "#/models/FS"}
      }
    },
    {
      "description":       "This       Compiled       release       includes       EI's       hi-level",
      "properties":                                                         {
        "ocid":                                                            "",
        "compiledRelease":                          {"$ref":                          "#/models/EI"}
      }
    }
  ]
}
```

Contracting Process

CP is a overarching hi-level common data and the budget of procurement

Statechart statuses

Contracting process statuses

The *status f*ield is used to indicate the current status of a tender process.

Title	Description
`planning`	A future contracting process is being considered
`planned`	A contracting process is scheduled, but is not yet taking place
`active`	A tender process is currently taking place
`cancelled`	The tender process has been cancelled.
`unsuccessful`	The tender process was unsuccessful.
`complete`	The tender process is complete.
`withdrawn`	No further information on this process is available under this ocid.

Contracting process status details

The *statusDetails* field is used to indicate the current state of a tender process.

Title	Description
`preselection`	This phase is established to achieve the aims of pre-selection step
`preselected`	The aims of preselection step are achieved within this hase
`prequalification`	This phase is established to achieve the aims of pre-qualification step
`prequalified`	The aims of pre-qualification step are achieved within this hase
`evaluation`	This phase is established to achieve the aims of evaluation step
`evaluated`	The aims of evaluation step are achieved within this hase
`execution`	This phase describes an implementation of the concluded contract
`suspended`	This phase is suspended due to applied business logic

Query Model

CP described as Release Package according to the following scheme:

Title, description	Type
`ocid` A globally unique identifier for this Contracting Process	string
`id` An identifier for this particular release of information for CP	string
`date` The date this information is released	date-time
`tag` A value that identifies the nature of the release being made	string
`initiationType`	string

String specifying the type of initiation process used for this contract		
planning.rationale The rationale for the procurement provided in free text.		string
planning.budget.description A short free text description of the budget source		string
planning.budget.amount Object described the sum of funds specified by CA this CP See Value		object
planning.budget.isEuropeanUnionFunded True/False field to indicate whether this procurement is related to a project and/or programme financed by EU		boolean
planning.budget.budgetBreakdown Detailed budget breakdown to be expressed, covering multiple budget sources and multiple periods	Array	object
tender.id An identifier for this CP		string
tender.title Title for this CP		string
tender.description Description for this CP		string
tender.classification The primary classification for the tender. Uses CPV Codelist See Classification		object
tender.status The current status of the CP based on the table of the CP statuses		string
tender.statusDetails The current status of the CP based on the table of the CP statuses		string
tender.value The total upper estimated value of this CP See Value		object
tender.procurementMethod Specify tendering method against the procurementMethods codelist		string
tender.procurementMethodDetails Additional detail on the procurement method used.		string
tender.procurementMethodRationale Rationale of procurement method		string
tender.procurementMethodAdditionalInfo Additional information about the procurement method		string
tender.mainProcurementCategory The primary category describing the object of this contracting process from the procurementCategory codelist		string
tender.additionalProcurementCategories Additional categories which describe the subject of this CP, from the extendedProcurementCategory codelist		string

tender.hasEnquiries		boolean
A true/false field to indicate whether any enquiries were received during the tender process		
tender.eligibilityCriteria		string
A description of any eligibility criteria for potential suppliers.		
tender.submissionLanguages	Array	string
Language(s) in which tenderers may submit		
tender.amendments	Array	object
A tender amendment is a formal change to the CN See Amendment		
tender.contractPeriod		object
The period over which the contract is estimated or required to be active See Period		
tender.acceleratedProcedure.isAcceleratedProcedure		boolean
A True/False field to indicate whether an accelerated procedure has been used for this procurement		
tender.designContest.serviceContractAward		boolean
A True/False field to indicate whether a service contract will be awarded to the winner(s) of the design contest.		
tender.electronicWorkflows.useOrdering		boolean
A True/False field to indicate if electronic ordering will be used.		
tender.electronicWorkflows.usePayment		boolean
A True/False field to indicate if electronic payment will be used.		
tender.electronicWorkflows.acceptInvoicing		boolean
A True/False field to indicate if electronic invoicing will be accepted.		
tender.jointProcurement.isJointProcurement		boolean
A True/False field to indicate if this is a joint procurement or not.		
tender.procedureOutsourcing.procedureOutsourced		boolean
A True/False field to indicate whether the procurement procedure has been outsourced		
tender.framework.isAFramework		boolean
A True/False field to indicate whether a framework agreement has been established as part of this procurement		
tender.hasDynamicPurchasingSystem		boolean
A True/False field to indicate whether a Dynamic Purchasing System has been set up.		
tender.legalBasis		string
The legal basis of the tender		
relatedProcesses	Array	object
Objects described information about related funding (FSs), EI and other related sub-processes See RelatedProcess		
parties	Array	object
Object described information about Entities involved into this CP See Organization		

Contracting Process details

Designed as a separate Release Package with detailed information of specific contracting process related to the hi-level common data of the process (CP). Depending of CAs procurement strategy such RElease Package may include data in different consistencies:

- *Periodic Notice* (PN) - this kind of notice means that CA have an intention to conduct a tender in some date during procurement year
- *Prior Information Notice* (PIN) - this kind of notice means that CA have an intention to conduct a tender in particular date (date, month or quarter) of procurement year
- *Contract Notice for evaluation* (EV) or pre-selection (PS) - this kind of notice means that the CA has an intention to conduct a t”Cender and start to collect bids right after publication.

Contract Notice (CN)

‘CN’ is a model describes a specific contract process - the procurement procedure announced by the CA. Depending on the type of procedure, method of procurement, geography and the legal basis, the attribute composition of the model can be adjusted, but its general logic remains unchanged for all types of procedures.

Transport model

CN described as Record Package consists of four (at least) or more Release Packages:

1. ‘cnParent’ (hi-level common data of Contract Notice and a budget of procurement)
2. ‘cnDetails’ with all other CN data (depends on the starting point (stage): PN, PIN or CfC (PS/PQ/EV)
3. ‘eiDetails’ (hi-level common data of used EI)
4. at least one ‘fs’ (detailed information about used FS)

```
{
  "uri":                                                                    "",
  "version":                                                                "",
  "extensions":                                                             [],
  "publisher":                                                              {},
  "license":                                                                "",
  "publishedDate":                                                          "",
  "packages":                                                                 [
    "uri                  of                 EI                release                package",
    "uri(s)               of               FS(s)               release             package(s)",
    "uri             of            CN            hi-level           release            package",
    "uri             of            CN             details           release             package"
  ],
```

```
  "records":                                                                    [
    {
      "description":       "This       Compiled       release       includes       CN's       hi-level",
      "properties":                                                                {
        "ocid":                                                                  "",
        "compiledRelease":                          {"$ref":                     "#/models/cnParent"}
      }
    },
    {
      "description":       "This       compiled       release       includes       CN's       details",
      "properties":                                                                {
        "ocid":                                                                  "",
        "compiledRelease":                          {"$ref":                     "#/models/cnDetails"}
      }
    },
    {
      "description":       "This       compiled       release       includes       FS       details",
      "properties":                                                                {
        "ocid":                                                                  "",
        "compiledRelease":                          {"$ref":                     "#/models/fs"}
      }
    },
    {
      "description":       "This       Compiled       release       includes       EI's       hi-level",
      "properties":                                                                {
        "ocid":                                                                  "",
        "compiledRelease":                          {"$ref":                     "#/models/ei"}
      }
    }
  ]
}
```


contractNoticeParent (cnParent) Query Model

Schema

ocid A globally unique identifier for this Contracting Process	string
id An identifier for this particular release of information for CP	string
date The date this information is released	date-time
tag A value from the releaseTag codelist that identifies the nature of the release being made	string
initiationType String specifying the type of initiation process used for this contract, taken from the initiationType codelist	string
planning.rationale The rationale for the procurement provided in free text.	string
planning.budget.description A short free text description of the budget source	string
planning.budget.amount	object

Object described the sum of funds specified by CA this CP
See Value

planning.budget.isEuropeanUnionFunded boolean

True/False field to indicate whether this procurement is related to a project and/or programme financed by EU

planning.budget.budgetBreakdown array object

Detailed budget breakdown to be expressed, covering multiple budget sources and multiple periods
See BudgetBreakdown

tender.id string

An identifier for this CP

tender.title string

Title for this CP

tender.description string

Description for this CP

tender.classification object

The primary classification for the tender. Uses CPV Codelist
See Classification

tender.status string

The current status of the CP based on the tenderStatus codelist

tender.statusDetails string

The current status of the CP based on the tenderStatusDetails codelist

tender.value object

The total upper estimated value of this CP
See Value

tender.procurementMethod string

Specify tendering method against the procurementMethods codelist

tender.procurementMethodDetails string

Additional detail on the procurement method used.

tender.procurementMethodRationale string

Rationale of procurement method

tender.procurementMethodAdditionalInfo string

Additional information about the procurement method

tender.mainProcurementCategory string

The primary category describing the object of this contracting process from the procurementCategory codelist

tender.additionalProcurementCategories string

Additional categories which describe the subject of this CP, from the extendedProcurementCategory codelist

tender.hasEnquiries boolean

A true/false field to indicate whether any enquiries were received during the tender process

tender.eligibilityCriteria string

A description of any eligibility criteria for potential suppliers.

tender.submissionLanguages	array	string
Language(s) in which tenderers may submit, drawn from the submissionLanguages codelist		
tender.amendments	array	object
A tender amendment is a formal change to the CN See Amendment		
tender.contractPeriod		object
The period over which the contract is estimated or required to be active See Period		
tender.acceleratedProcedure.isAcceleratedProcedure		boolean
A True/False field to indicate whether an accelerated procedure has been used for this procurement		
tender.designContest.serviceContractAward		boolean
A True/False field to indicate whether a service contract will be awarded to the winner(s) of the design contest.		
tender.electronicWorkflows.useOrdering		boolean
A True/False field to indicate if electronic ordering will be used.		
tender.electronicWorkflows.usePayment		boolean
A True/False field to indicate if electronic payment will be used.		
tender.electronicWorkflows.acceptInvoicing		boolean
A True/False field to indicate if electronic invoicing will be accepted.		
tender.jointProcurement.isJointProcurement		boolean
A True/False field to indicate if this is a joint procurement or not.		
tender.procedureOutsourcing.procedureOutsourced		boolean
A True/False field to indicate whether the procurement procedure has been outsourced		
tender.framework.isAFramework		boolean
A True/False field to indicate whether a framework agreement has been established as part of this procurement		
tender.hasDynamicPurchasingSystem		boolean
A True/False field to indicate whether a Dynamic Purchasing System has been set up.		
tender.legalBasis		string
The legal basis of the tender based on the legalBasis codelist		
tender.procuringEntity		object
Link to the Organization operating this Process including a set of evaluation panel if declared by PE See OrganizationReference		
relatedProcesses	array	object
Objects described information about related funding (FSs), EI and other related sub-processes See RelatedProcess		
parties	array	object
Object described information about Entities involved into this CP See Organization		

'cnParent' described as Release Package and can be described in two ways:

‘cnParent’ as Compiled Release


```
{
  "ocid": "",
  "id": "",
  "date": "",
  "tag": "compiled",
  "initiationType": "tender",
  "planning": {"$ref": "#/definitions/Planning"},
  "tender": {
    "id": "",
    "title": "",
    "description": "",
    "classification": {"$ref": "#/definitions/Classification"},
    "status": "",
    "statusDetails": "",
    "value": {"$ref": "#/definitions/Value"},
    "procurementMethod": "",
    "procurementMethodDetails": "",
    "procurementMethodRationale": "",
    "procurementMethodAdditionalInfo": "",
    "mainProcurementCategory": "",
    “procuringEntity”: {
        “id”:””,
        “name”: “”},
    "additionalProcurementCategories": [],
    "hasEnquiries": "",
    "eligibilityCriteria": "",
    "contractPeriod": {"$ref": "#/definitions/Period"},
    "acceleratedProcedure": {"$ref": "#/definitions/AcceleratedProcedure"},
    "designContest": {"$ref": "#/definitions/DesignContest"},
    "electronicWorkflows": {"$ref": "#/definitions/ElectronicWorkflows"},
    "jointProcurement": {"$ref": "#/definitions/JointProcurement"},
    "procedureOutsourcing": {"$ref": "#/definitions/ProcedureOutsorcing"},
    "framework": {"$ref": "#/definitions/Framework"},
    "dynamicPurchasingSystem": {"$ref": "#/definitions/DynamicPurchasingSystem"},
    "legalBasis": ""
  },
  "parties": [{"$ref": "#/definitions/Organization"}],
  "relatedProcesses": [
    {"$ref": "#/definitions/RelatedProcess"}
  ]
}
```


‘cnParent’ as Release Package


```
{
  "version": "",
  "uri": "",
  "publishedDate": "",
  "publisher": {
    "$ref": "#/definitions/Publisher"
  },
  "license": "",
  "publicationPolicy": "",
  "releases": [
    "$ref": "#/models/cnParent"
  ]
```

```
}
```

contractNoticeDetails (cnDetails) Query Model

Scheme

ocid	string
A globally unique identifier for this part of Contracting Process	
id	string
An identifier for this particular release of information for part of CP	
date	date-time
The date this information is released	
tag	string
A value from the releaseTag codelist that identifies the nature of the release being made	
initiationType	string
String specifying the type of initiation process used for this part of CP, taken from the initiationType codelist	
purposeOfNotice.isACallForCompetition	boolean
A True/False field to indicate whether this notice is a call for competition	
tender.id	string
An identifier for this part of CP	
tender.title	string
Title for this part of CP	
tender.description	string
Description for this part of CP	
tender.awardCriteria	string
Specify the award criteria for the procurement, using the award criteria codelist	
tender.awardCriteriaDetails	string
Specify the award criteria details for the procurement, using the award criteria details codelist	
tender.conversions	array
Conversions to be applied for the criteria See Conversion	
tender.criteria	array
Criteria required for the tenderers See Criterion	
tender.status	string
The current status of this part of CP based on the tenderStatus codelist	
tender.statusDetails	string
The current status of this part of CP based on the tenderStatusDetails codelist	
tender.enquiryPeriod	
The period during which potential bidders may submit questions and requests for clarification	

See Period		
tender.hasEnquiries		boolean
A true/false field to indicate whether any enquiries were received during the tender process.		
tender.enquiries	array	object
List of received enquiries and comment from PE See Enquiry		
tender.standStillPeriod		
See Period		
tender.lotGroups.optionToCombine		boolean
True/False value indicates the CA reserves the right to combine the lots in this group when awarding a contract		
tender.lots	array	object
A tender process is divided into lots. See Lot		
tender.items	array	object
The goods and services to be purchased See Item		
tender.requiresElectronicCatalogue		boolean
True/False value indicates whether bids must include an electronic catalogue.		
tender.submissionMethod		string
Specify the method by which bids must be submitted using the submission method codelist		
tender.submissionMethodRationale		string
A value from the submissionValueRationale codelist that identifies the rationale where electronic submission method is not to be allowed		
tender.submissionMethodDetails		string
Any detailed or further information on the submission method.		
tender.tenderPeriod		object
The period when this part of CP will be open for submission See Period		
tender.procurementMethodModalities		string
The modalities of the procurement method indicated with Method Modalities codelist		
tender.auctionPeriod		object
General period of all auctions scheduled under specific contracting process See Period		
tender.electronicAuctions	array	object
See ElectronicAuctions		
tender.documents	array	object
All documents and attachments related to the tender, including any notices. See the documentType codelist See Document		

bids.details	array	object
An array of bids, providing information on the bidders, bid status, bid values and related documents. See Bid		
bids.statistics	array	object
Summary statistics on the number and nature of bids received. See BidStatistic		
awards	array	object
A list of An award for the given procurement See Award		
tender.awardPeriod		object
The period for adjudication and selection of the contract award See Period		
contracts	array	object
Information regarding the signed contract between the buyer and supplier(s) See Contract		
relatedProcesses	array	object
Objects described information about related funding (FSs), EI and other related sub-processes See RelatedProcess		
parties	array	object
Object described information about Entities involved into this part of CP See Organization		

Options

Notification options

'cnDetails' described as Release Package related with at least 'cnParent' - hi-level common data of the process. Depending of CAs procurement strategy 'cnDetails' may include data in three different consistencies:

- *Planning Notice* (planningNotice) - using this kind of notice means that CA have an intention to conduct a tender in some date during procurement year
- *Prior Information Notice* (priorNotice) - using this kind of notice means that CA have an intention to conduct a tender in particular date (date, month or quarter) of procurement year
- *Contract Notice* (contractNotice) - using this kind of notice means that CA have an intention to conduct a tender and start to collect a bids right after publication date

In any case budget of future tender MUST be defined and indicated

Electronic reverse auction as an option

Contracting authorities may use electronic auctions, in which new prices, revised downwards, and/or new values concerning certain elements of tenders are presented. For this purpose, contracting authorities shall structure the electronic auction as a repetitive electronic process and include it into request for Contract Notice (CN only):

```
{
  "tender":{
    "procurementMethodModalities": [
      "electronicAuction"
```

```
    ],
    "electronicAuctions":                                                    {
      "details":                                                             [
        {
          "id":                                                            "",
          "relatedLot":                                                    "",
          "electronicAuctionModalities":                                     [
            {
              "eligibleMinimumDifference": {
                "amount": 1000.00,
                "currency": "MDL"
              }
            }
          ]
        }
      ]
    }
  }
}
```

If 'electronicAuction' included into CN creation payload as one of used procurementMethodModality, 'eligibleMinimumDifference' must be specified for each lot under this contracting process. The 'currency' must be equal to 'tender.value.currency' defined in the cnParent

planningNotice

Planning notice - is an announcement of intention to conduct a tender. On this stage just hi-level params of future procedure should be indicated including estimated period of start (month or quarter). Specification of object of procurement, detailed award and non-price criteria etc could be announced with Prior Information Notice or Contract Notice. 'planningNotice' described as Release Package and can be described in two ways:

'planningNotice' as Compiled Release

```
{
  "ocid": "",
  "id": "",
  "date": "",
  "tag": "planning",
  "initiationType": "tender",
  "hasPreviousNotice" : false,
  "purposeOfNotice": {
    "isACallForCompetition": false
  },
  "tender": {
    "id": "",
    "title": "",
    "description": "",
    "status": "",
    "statusDetails": "",
    "requiresElectronicCatalogue": false,
    "submissionMethod": [""],
    "submissionMethodRationale": [""],
    "submissionMethodDetails": "",
    "tenderPeriod": {
      "startDate": ""
    },
    "documents": [
      { "$ref": "#/definitions/Document" }
    ],
    "lotGroups": [
      { "optionToCombine": false}
    ],
    "lots": [
      { "$ref": "#/definitions/Lot" }
    ],
    "items": [
      { "$ref": "#/definitions/item" }
    ]
  },
  "relatedProcesses": [
    { "$ref": "#/definitions/RelatedProcess" }
  ]
}
```

'planningNotice' as Release Package

```
{
  "version": "",
  "uri": "",
  "publishedDate": "",
  "publisher": {
```

```
    "$ref":                                                    "#/definitions/Publisher"
  },
  "license":                                                                          "",
  "publicationPolicy":                                                                "",
  "releases":              [              "$ref":              "#/models/planningNotice"              ]
}
```

priorNotice

Prior Information Notice - publication of this kind of notice means that CA has an intention to conduct a tender in known date of procurement year. 'priorNotice' described as Release Package and can be described as:

'priorNotice' as Compiled Release

```
{
  "ocid":                                                                            "",
  "id":                                                                              "",
  "date":                                                                            "",
  "tag":                                                                        "planned",
  "initiationType":                                                              "tender",
  "hasPreviousNotice":                                                             false,
  "purposeOfNotice":                                                                  {
    "isACallForCompetition":                                                        true
  },
  "tender":                                                                           {
    "id":                                                                            "",
    "title":                                                                         "",
    "description":                                                                   "",
    "status":                                                                        "",
    "statusDetails":                                                                 "",
    "awardCriteria":                                                                 "",
    "requiresElectronicCatalogue":                                                 false,
    "submissionMethod":                                                              "",
    "submissionMethodRationale":                                                     "",
    "submissionMethodDetails":                                                       "",
    "tenderPeriod":                                                                    {
      "startDate":                                                                   ""
    },
    "documents":                                                                      [
      {
        "$ref":                                                  "#/definitions/Document"
      }
    ],
    "lotGroups":                                                                      [
      {"$ref":                                                 "#/definitions/LotGroup"}
    ],
    "lots":                                                                           [
      {"$ref":                                                      "#/definitions/Lot"}
    ],
    "items":                                                                          [
      {"$ref":                                                    "#/definitions/item"}
    ]
  },
  "relatedProcesses":                                                                 [
    {                    "$ref":                   "#/definitions/RelatedProcess"                    }
  ]
}
```

'priorNotice' as Release Package

```
{
  "version":                                                                    "",
  "uri":                                                                        "",
  "publishedDate":                                                              "",
  "publisher":                                                                   {
    "$ref":                                                   "#/definitions/Publisher"
  },
  "license":                                                                    "",
  "publicationPolicy":                                                          "",
  "releases":                                                                    [
    "$ref":                                                     "#/models/priorNotice"
  ]
}
```

contractNotice

Contract Notice - publication of this kind of notice means that CA has an intention to conduct a tender and start to collect bids right after publication. 'contractNotice' described as Release Package and can be described as:

'contractNotice' as Compiled Release

```
{
  "ocid":                                                                      "",
  "id":                                                                        "",
  "date":                                                                      "",
  "tag":                                                                  ["tender"],
  "initiationType":                                                            "",
  "hasPreviousNotice":                                                        true,
  "purposeOfNotice":                          {"isACallForCompetition":       true},
  "tender":                                                                     {
    "id":                                                                      "",
    "title":                                                                   "",
    "description":                                                             "",
    "status":                                                                  "",
    "statusDetails":                                                           "",
    "awardCriteria": "",
    "awardCriteriaDetails": "",
    "criteria": [{"$ref": "#/definitions/Criterion"}],
    "conversions": [{"$ref": "#/definitions/Conversion}],
    "requiresElectronicCatalogue":false,
    "submissionMethod":                                                        "",
    "submissionMethodRationale":                                               "",
    "submissionMethodDetails":                                                 "",
    "hasEnquiries":                                                          false,
    "enquiryPeriod":                 {          "$ref":          "#/definitions/Period"},
    "lotGroups":                              [{"optionToCombine":                 ""}],
    "lots":                                                                     [
      {"$ref":                                                  "#/definitions/Lot"}
    ],
    "items":                                                                    [
      {"$ref":                                                 "#/definitions/item"}
    ],
    "tenderPeriod":                {"$ref":          "#/definitions/Period"          },
    "documents":                                                                [
      {"$ref":                                                "#/definitions/Document"}
    ]
```

```
  },
  "parties":                                                                    {
    "items":                        {"$ref":                        "#/definitions/OrganizationReference"}
  },
  "relatedProcesses":                                                           [
    {                    "$ref":                    "#/definitions/RelatedProcess"                    }
  ]
}
```

'contractNotice' as Release Package

```
{
  "version":                                                                    "",
  "uri":                                                                        "",
  "publishedDate":                                                              "",
  "publisher":                                                                  {
    "$ref":                                                  "#/definitions/Publisher"
  },
  "license":                                                                    "",
  "publicationPolicy":                                                          "",
  "releases":                                                                   [
    "$ref":                                                  "#/models/contractNotice"
  ]
}
```

Command Model

To create new entity of CN standard POST data model should be used for preparation of POST-request for BPE. Such POST-request must be based on CN data-model and includes only required and optional data-fields. Command model for all three kinds of notices is same. But the sets of required fields are different:

Schema

planning.rationale		optional
planning.budget.description		required for: CNonPN/update CN
planning.budget.budgetBreakdown	array	required for: PN/PIN/CN
planning.budget.budgetBreakdown.id An identifier for this particular budget entry - OCID of used FS		required
planning.budget.budgetBreakdown.amount ***Validation rules:*** • *The same value of budget.budgetBreakdown.amount.currency has to be delivered for each budgetBreakdown* • *Value of budget.budgetBreakdown.amount.amount has to be equal or less than budget.amount.amount in used FS*		required
tender.title		required for: PN/PIN/CN
tender.description		required for: PN/PIN/CN
tender.awardCriteria		required for: CNonPN
tender.awardCriteriaDetails		required for: CNonPN
tender.procurementMethodAdditionalInfo		optional
tender.submissionLanguages	array	optional
tender.legalBasis		required for: PN/PIN/CN
tender.procuringEntity		required for: PN/PIN/CN optional for: CNonPN/updateCN

tender.criteria		optional for CN of Online Procedure
tender.conversions		optional for CN of Online Procedure
tender.tenderPeriod		required for: PN/PIN/CN
tender.tenderPeriod.startDate		required for PN/PIN
tender.tenderPeriod.endDate		required for CN of Online Procedure
tender.documents	array	optional

Validation rules:

- *The values of tender.documents.relatedLots of all delivered documents to be matched with all delivered tender.lots.id*

tender.procurementMethodRationale		optional
tender.procurementMethodModalities		optional
tender.electronicAuctions		required for CN if previous used (for online procedures only)
tender.electronicAuctions.details.id		
Temporary identifier for needed auction		
tender.electronicAuctions.details.relatedLot		
Temporary identifier of lot for which auction needs to be scheduled		
tender.electronicAuctions.details.electronicAuctionModalities		
tender.lots	array	required for: PIN/CN
tender.lot.id		required
Temporary identifier for this lot, such as a lot number.		
tender.lot.title		required
tender.lot.description		optional
tender.lot.internalId		optional
tender.lot.value		required

Validation rules:

- *tender.lot.value.currency in all added lots has to be equal to budget.budgetBreakdown.amount.currency*
- *the cumulative sum of all added tender.lot.value.amount has to be equal or less than the sum of all added budget.budgetBreakdown.amount.amount*

tender.lot.contractPeriod		required

Validation rules:

- *tender.lot.contractPeriod.startDate has to be earlier than Tender.lot.contractPeriod.endDate in one lot object & tender.lot.contractPeriod.startDate has to be later than tender.tenderPeriod.endDate*
- *earliest value among all added tender.lot.contractPeriod.startDate has to be earlier than values of all added planning.budget.budgetBreakdown.period.endDate*
- *the latest value among all added tender.lot.contractPeriod.endDate has to be later than values of all added planning.budget.budgetBreakdown.period.startDate*

tender.lot.placeOfPerformance		required

Validation rules:

- *Lots object have to include at least one lot*

tender.items	array	required for: PIN/CN
tender.item.id		required
Temporary identifier to reference the items by		
tender.item.description		optional
tender.item.relatedLots		required

Validation rules:

- *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*

tender.item.classification		required

Validation rules:

- *All added tender.Item.classification.id in all items coincide by first 3 figures in codes*
- *All added tender.Item.classification.id in all items coincide by first 3 figures of value of tender.classification.id in EI*

tender.item.additionalClassifications	array	optional
tender.item.quantity		required
tender.item.unit		required
tender.item.internalId		optional
tender.enquiryPeriod		required for CN
tender.enquiryPeriod.endDate		required (for online procedures only)

Examples

Planning Notice

```
{
  "planning": {
    "rationale": "",
    "budget": {
      "budgetBreakdown": [
        {
          "id": "",
          "amount": {}
        }
      ]
    }
  },
  "tender": {
    "title": "",
    "description": "",
    "legalBasis": "",
    "tenderPeriod": { "startDate": "" },
    "procuringEntity": { "$ref": "#/definitions/Organization" }
  }
}
```

Contract Notice on existing Planning Notice (CNonPN)

```
{
  "tender": {
    "title": "",
    "description": "",
    "awardCriteria": "",
    "enquiryPeriod": {
      "endDate": ""
    },
    "tenderPeriod": {
      "endDate": ""
    },
    "lots": [
      {
        "id": "",
        "title": "",
        "description": "",
        "value": { "$ref": "#/definitions/Value" },
        "contractPeriod": { "$ref": "#/definitions/Period" },
        "placeOfPerformance": { "$ref": "#/definitions/PlaceOfPerformance" }
      }
```

```
    ],
    "items": [ { "$ref": "#/definitions/Item" } ],
    "documents": [ { "$ref": "#/definitions/Document" } ],

    // if electronic auctions required, following set must be included

    "procurementMethodModalities": [
      "electronicAuction"
    ],
    "electronicAuctions": {
      "details": [
        {
          "id": "",
          "relatedLot": "",
          "electronicAuctionModalities": [
            {
              "eligibleMinimumDifference": {
                "amount": "",
                "currency": ""
              }
            }
          ]
        }
      ]
    }
  }
}
```

Updates

The update of existing entity of notice has several options of execution depending of type of notice and its status as well as intention to modify/include detailed description of nomenclature of subject of procurement.

Model for PN update

Planning Notice can be updated in planning status only - thus, before active process phase (e.g. EV) is launched under parent Contracting Process.

Scheme

planning.rationale	Can be amended as free text
tender.title	Can be amended as free text
tender.description	Can be amended as free text
tender.procurementMethodAdditionalInfo	Can be amended as free text
tender.tenderPeriod.startDate	Launch of start of submission can be postponed
tender.documents	Set of document can be updated with new elements or previously published documents can be updated with new versions
tender.procurementMethodRationale	Can be amended as free text
tender.lots	Set of lots can be added (if initial model did not include any lots) or updated with new elements by splitting existing one to two or more separate lots with the appropriate allocation of the initial budget for splitting lot. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*
Fields that can be amended	
tender.lot.title	Can be amended as free text
tender.lot.description	Can be amended as free text
tender.lot.contractPeriod	Period over which the contract is estimated or required to be active for this lot
Note that: ● *any updates of lot.contractPeriod will have an influence to common tender.contractPeriod* ● *any updates of lot.contractPeriod will be validated according to used FS and their planning.budget.budgetPeriod*	
tender.lot.placeOfPerformance	Can be amended as free text
tender.items	Set of items can not be changed but initially existing items can be amended with new values of following fields.Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.* ***Note that:*** ● *In case of exclusion on or more lots from update-request, all related items will stay in release package of CN with no connection.*
Fields that can be amended	
tender.item.description	Can be amended as free text
tender.item.relatedLots	Relation with one of initial or added lots can be changed
Validation rules: ● *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*	

Example


```
{
  "planning": {
    "rationale": ""
  },
  "tender": {
    "title": "",
    "description": "",
    "procurementMethodAdditionalInfo": "",
    "submissionLanguages": [],
    "procurementMethodRationale": "",
    "tenderPeriod": {
      "startDate": ""
    },
    "lots": [
      {
        "id": "",
        "title": "",
        "description": "",
        "value": {
          "$ref": "#/definitions/Value"
        },
        "contractPeriod": {
          "$ref": "#/definitions/Period"
        },
        "placeOfPerformance": {
          "$ref": "#/definitions/PlaceOfPerformance"
        }
      }
    ],
    "items": [
      {
        "$ref": "#/definitions/Item"
      }
    ],
    "documents": [
      {
        "$ref": "#/definitions/Document"
      }
    ]
  }
}
```

Model for CN update

Contract Notice can be updated during clarification of contracting process only

tender.title	Can be amended as free text
tender.description	Can be amended as free text
tender.enquiryPeriod.endDate	Deadline for clarification can be postponed. Submission period deadline will be rescheduled according to the rules relevant to the procedure under used legal basis
tender.tenderPeriod.endDate	Deadline for submission can be postponed considering minimum eligible duration according to used procedure - for online procedures only
tender.documents	Set of document can be updated with new elements or previously published documents can be updated with new versions

Validation rules:

- *All the document that exist in initially created CN must be present in the payload of update request*

tender.lots	Set of lots can be updated with new elements by splitting existing one to two or more separate lots with the appropriate allocation of the initial budget for splitting lot. Initially published lots can be switched to *status:cancelled by excluding of such lot from update.*

Note that:

- *In case of exclusion one or more lots from update-request, all related items will stay in release package of CN*

Fields that can be amended	
tender.lot.title	Can be amended as free text
tender.lot.description	Can be amended as free text
tender.lot.contractPeriod	Period over which the contract is estimated or required to be active for this lot

Note that:

- *any updates of lot.contractPeriod will have an influence to common tender.contractPeriod*
- *any updates of lot.contractPeriod will be validated according to used FS and their planning.budget.budgetPeriod*

tender.lot.placeOfPerformance	Can be amended as an object
tender.items	Set of items can not be changed but initially existing items can be amended with new values of following fields. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*

Note that:

- *In case of exclusion on or more lots from update-request, all related items will stay in release package of CN with no connection.*

Fields that can be amended	
tender.item.description	Can be amended as free text
tender.item.relatedLots	Relation with one of initial or added lots can be changed

Validation rules:

- *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*

tender.procuringEntity

Note that:

- *In case of exclusion on or more lots from update-request, all related items will stay in release package of CN with no connection.*

Fields that can be amended	
tender.procuringEntity.persones	New persons can be added

Example

```
{
  "tender": {
    "title": "",
    "description": "",
    "enquiryPeriod": {
      "endDate": ""
    },
    "tenderPeriod": {
      "endDate": ""
    },
    "lots": [
      {
        "id": "",
        "title": "",
        "description": "",
        "value": {
          "$ref": "#/definitions/Value"
        },
        "contractPeriod": {
          "$ref": "#/definitions/Period"
        },
        "placeOfPerformance": {
          "$ref": "#/definitions/PlaceOfPerformance"
        }
      }
    ],
    "items": [
      {
        "$ref": "#/definitions/Item"
      }
    ],
    "documents": [
      {
        "$ref": "#/definitions/Document"
      }
    ],
    "procuringEntity": {
      "id": "",
      "persones":[
        {
          "$ref":"#/definitions/Persone"
        }
      ]
    }
  }
}
```

See how to work CN entity in Tutorial

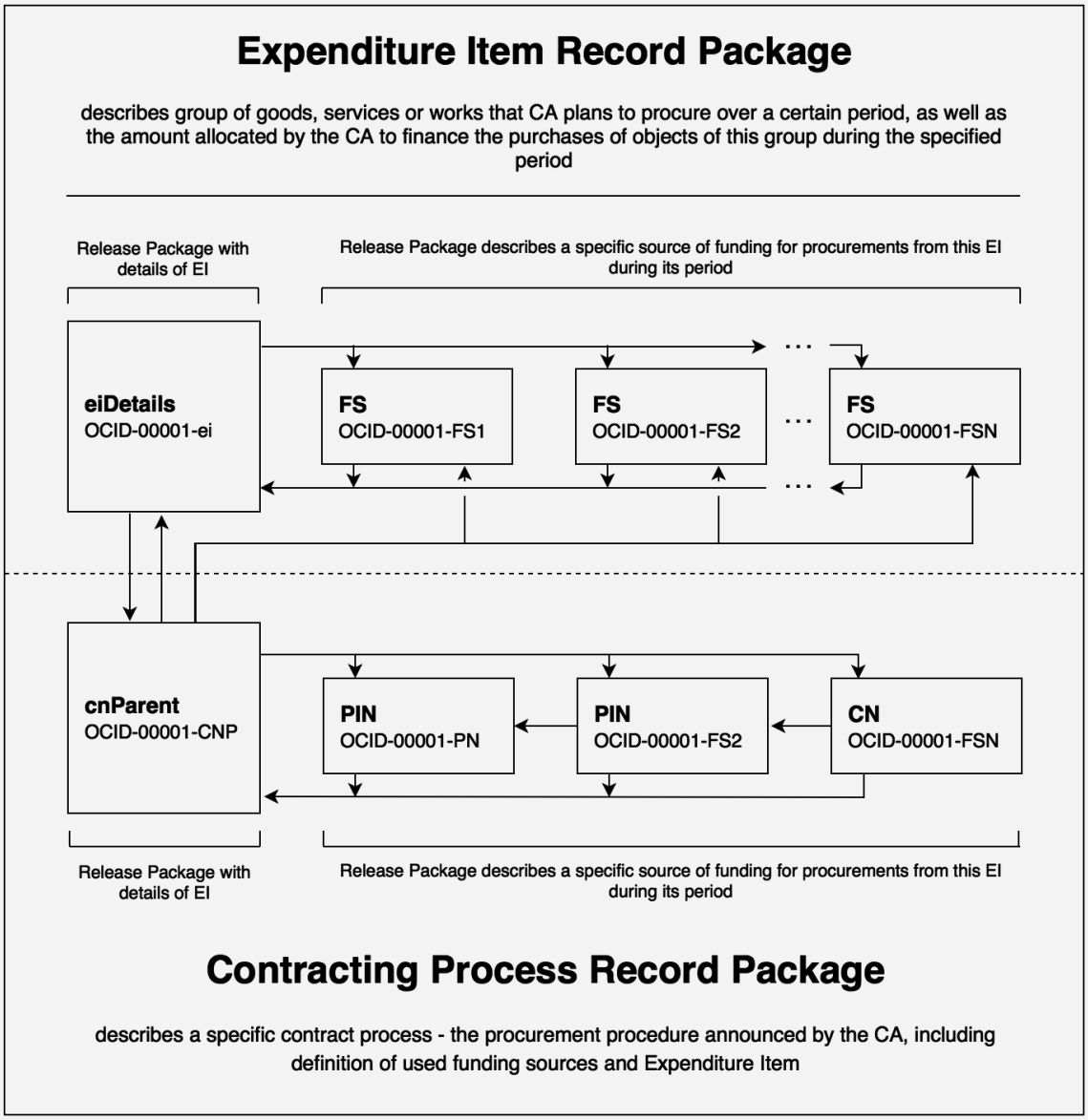


Figure 6.2. Scheme of Contracting Process description using EI, FS, cnParent, PN, PIN and CN models

4.2.1.3Planning

The planning section is used to describe the financial background to a contracting process. This include details of the budget from which funds are drawn or related budget-projects for this contracting process.

Periodic notice

Periodic notice - is an announcement of intention to conduct a tender. On this stage just hi-level attributes and values of future procedure should be indicated including estimated period of start

of submission (month or quarter). Specification of object of procurement, detailed award and non-price criteria etc could be announced with Prior Information Notice or Contract Notice.

'PN' data-model

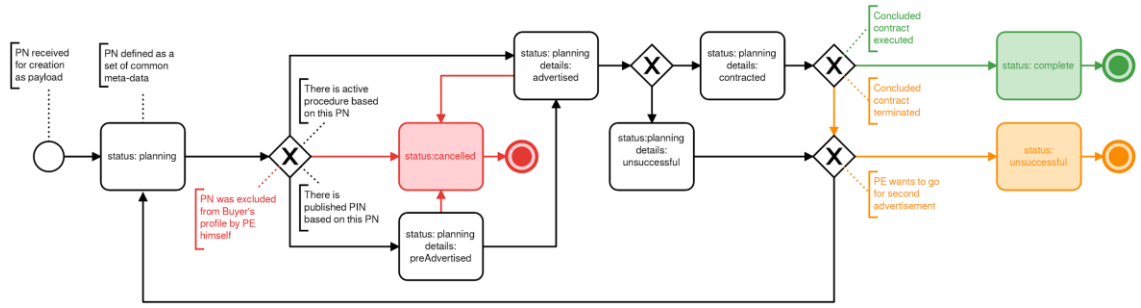


Figure 7. Statechart diagram for PN (go to https://goo.gl/VR2VW5)

Statechart statuses

Periodic Notice statuses

The *status f*ield is used to indicate the current status of a tender process.

Title	Description
`Planning`	A future contracting process is being considered
`Planned`	A contracting process is scheduled, but is not yet taking place
`Active`	A tender process is currently taking place
`Cancelled`	The tender process has been cancelled.
`unsuccessful`	The tender process was unsuccessful.
`Complete`	The tender process is complete.
`Withdrawn`	No further information on this process is available under this ocid.

Periodic Notice status details

The *statusDetails* field is used to indicate the current state of a tender process.

Title	Description
`preAdvertised`	There is no yet Contract Notice under this PN
`Advertised`	There is a Contract Notice under this PN
`Contracted`	The Contracting process established under this PN is contracted
`unsuccessful`	The Contracting process established under this PN is unsuccessful

Query model

Title, description	Type
`Ocid` A globally unique identifier for this part of Contracting Process	string
`Id` An identifier for this particular release of information for part of CP	string

Date		date-time
The date this information is released		
Tag		string
A value that identifies the nature of the release being made		
initiationType		string
String specifying the type of initiation process used for this part of CP		
purposeOfNotice.isACallForCompetition		boolean
A True/False field to indicate whether this notice is a call for competition		
tender.id		string
An identifier for this part of CP		
tender.title		string
Title for this part of CP		
tender.description		string
Description for this part of CP		
tender.status		string
The current status of this part of CP based on the table of statuses		
tender.statusDetails		string
The current status of this part of CP based on the table of statuses		
tender.lotGroups.optionToCombine		boolean
True/False value indicates the CA reserves the right to combine the lots in this group when awarding a contract		
tender.lots	array	object
A tender process is divided into lots.		
tender.items	array	object
The goods and services to be purchased		
tender.procurementMethodModalities		string
The modalities of the procurement method		
tender.documents	array	object
All documents and attachments related to the tender, including any notices		
relatedProcesses	array	object
Objects described information about related funding (FSs), EI and other related sub-processes		
Parties	array	object
Object described information about Entities involved into this part of CP		

Command model

To create new entity of PN standard POST data model should be used for preparation of POST-request for BPE.

Title	Description	Obligation

`planning.rationale`	Has to be specified as an object according to definition	optional
`planning.budget.budgetBreakdown`	Has to be specified as an object according to definition	required
Element attributes		
`planning.budget.budgetBreakdown.id`	Has to be specified according to budget classification	required
`planning.budget.budgetBreakdown.amount`	Has to be specified as an object according to definition	required

Validation rules:

- *The same value of budget.budgetBreakdown.amount.currency has to be delivered for each budgetBreakdown*
- *Value of budget.budgetBreakdown.amount.amount has to be equal or less than budget.amount.amount in used FS*

`tender.title`	Has to be specified as a free text	required
`tender.description`	Has to be specified	required
`tender.procurementMethodAdditionalInfo`	Has to be specified	optional
`tender.submissionLanguages`	Has to be specified	optional
`tender.legalBasis`	Has to be specified	required
`tender.procuringEntity`	Has to be specified	optional
`tender.tenderPeriod`	Has to be specified	required
Element attributes		
`tender.tenderPeriod.startDate`	Has to be indicatively specified	required
`tender.tenderPeriod.endDate`		optional
`tender.documents`	Has to be specified	optional

Validation rules:

- *The values of tender.documents.relatedLots of all delivered documents to be matched with all delivered tender.lots.id*

`tender.procurementMethodRationale`	Has to be specified	optional
`tender.lots`	Has to be specified	optional
Element attributes		
`tender.lots.id`	Has to be specified	required
`tender.lots.title`	Has to be specified	required
`tender.lots.description`	Has to be specified	optional
`tender.lots.value`	Has to be specified	required

Validation rules:

- *tender.lot.value.currency in all added lots has to be equal to budget.budgetBreakdown.amount.currency*
- *the cumulative sum of all added tender.lot.value.amount has to be equal or less than sum of all added budget.budgetBreakdown.amount.amount*

`tender.lots.contractPeriod`	Has to be specified	required

Validation rules:

- *tender.lot.contractPeriod.startDate has to be earlier than Tender.lot.contractPeriod.endDate in one lot object & tender.lot.contractPeriod.startDate has to be later than tender.tenderPeriod.endDate*
- *earliest value among all added tender.lot.contractPeriod.startDate has to be earlier than values of all added planning.budget.budgetBreakdown.period.endDate*
- *the latest value among all added tender.lot.contractPeriod.endDate has to be later than values of all added planning.budget.budgetBreakdown.period.startDate*

`tender.lots.placeOfPerformance`		required

Validation rules:

- *Lots object have to include at least one lot*

`tender.items`	Has to be specified	optional
Element attributes		

`tender.items.id`	Has to be specified	required
`tender.items.description`	Has to be specified	optional
`tender.items.relatedLots`	Has to be specified	required
Validation rules: • *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*		
`tender.item.classification`	Has to be specified	required
Validation rules: • *All added tender.Item.classification.id in all items coincide by first 3 figures in codes* • *All added tender.Item.classification.id in all items coincide by first 3 figures of value of tender.classification.id in EI*		
`tender.items.additionalClassifications`	Has to be specified	optional
`tender.items.quantity`	Has to be specified	required
`tender.items.unit`	Has to be specified	required

Update model

Periodic Notice can be updated in planning status only - thus, before active process phase (e.g. EV) is launched under parent Contracting Process.

Title	Description	Obligation
`planning.rationale`	Can be amended as free text	Optional
`tender.title`	Can be amended as free text	Optional
`tender.description`	Can be amended as free text	Optional
`tender.tenderPeriod.startDate`	Launch of start of submission can be postponed	
`tender.documents`	Set of document can be updated with new elements or previously published documents can be updated with new versions	
`tender.lots`	Set of lots can be added (if initial model did not include any lots) or updated with new elements by splitting existing one to two or more separate lots with the appropriate allocation of the initial budget for splitting lot. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*	
Fields that can be amended		
`tender.lots.title`	Can be amended as free text	
`tender.lots.description`	Can be amended as free text	
`tender.lots.contractPeriod`	Period over which the contract is estimated or required to be active for this lot	
Note that: *any updates of lot.contractPeriod will have an influence to common tender.contractPeriod* *any updates of lot.contractPeriod will be validated according to used FS and their planning.budget.budgetPeriod*		
`tender.lots.placeOfPerformance`	Can be amended as free text	
`tender.items`	Set of items can not be changed but initially existing items can be amended with new values of following fields.Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*	

Note that:
In case of exclusion on or more lots from update-request, all related items will stay in release package of CN with no connection.

Fields that can be amended	
`tender.items.description`	Can be amended as free text

`tender.items.relatedLots`	Relation with one of initial or added lots can be changed

Validation rules:

Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id

Prior Information Notice

Prior Information Notice - publication of this kind of notice means that CA has an intention to conduct a tender at a known date in a procurement year.

'PIN' data-model

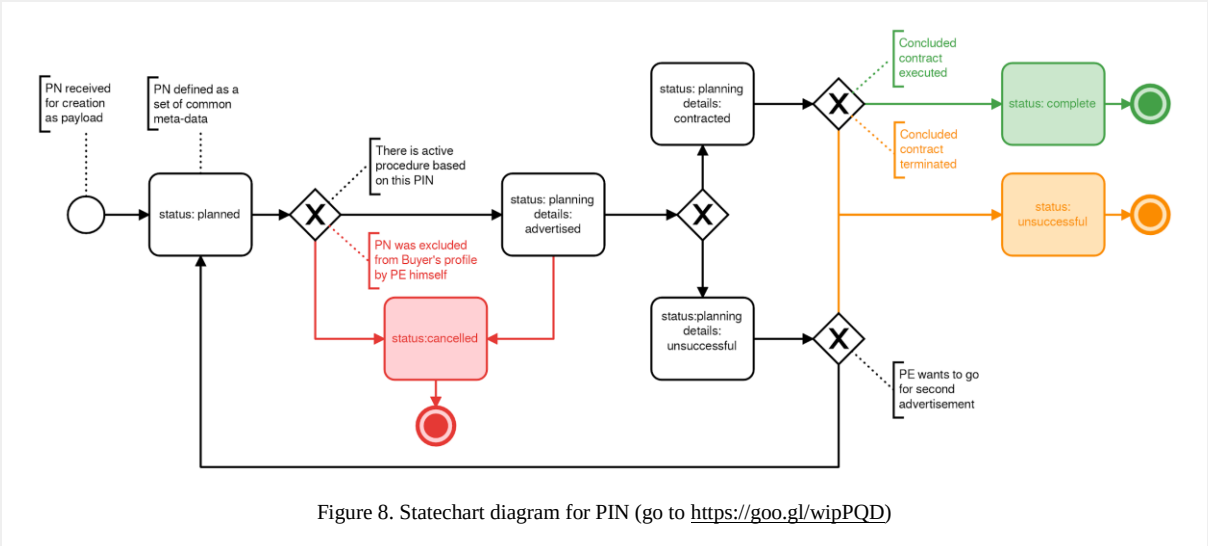


Figure 8. Statechart diagram for PIN (go to https://goo.gl/wipPQD)

Statechart statuses

Prior Information Notice statuses

The *status f*ield is used to indicate the current status of a tender process.

Title	Description
`Planning`	A future contracting process is being considered
`Planned`	A contracting process is scheduled, but is not yet taking place
`Active`	A tender process is currently taking place
`Cancelled`	The tender process has been cancelled.
`unsuccessful`	The tender process was unsuccessful.
`Complete`	The tender process is complete.
`Withdrawn`	No further information on this process is available under this ocid.

Prior Information Notice status details

The *statusDetails* field is used to indicate the current state of a tender process.

Title	Description
`preAdvertised`	There is no yet Contract Notice under this PN
`Advertised`	There is a Contract Notice under this PN
`Contracted`	The Contracting process established under this PN is contracted
`unsuccessful`	The Contracting process established under this PN is unsuccessful

Query model

Title, description		Type
Ocid		string
A globally unique identifier for this part of Contracting Process		
Id		string
An identifier for this particular release of information for part of CP		
Date		date-time
The date this information is released		
Tag		string
A value that identifies the nature of the release being made		
initiationType		string
String specifying the type of initiation process used for this part of CP		
purposeOfNotice.isACallForCompetition		boolean
A True/False field to indicate whether this notice is a call for competition		
tender.id		string
An identifier for this part of CP		
tender.title		string
Title for this part of CP		
tender.description		string
Description for this part of CP		
tender.awardCriteria		object
Specify the award criteria for the procurement		
tender.status		string
The current status of this part of CP based on the table of statuses		
tender.statusDetails		string
The current status of this part of CP based on the table of statuses		
tender.lotGroups.optionToCombine		boolean
True/False value indicates the CA reserves the right to combine the lots in this group when awarding a contract		
tender.lots	array	object
A tender process is divided into lots.		
tender.items	array	object
The goods and services to be purchased		
tender.requiresElectronicCatalogue		boolean
True/False value indicates whether bids must include an electronic catalogue.		
tender.submissionMethod		string
Specify the method by which bids must be submitted		
tender.submissionMethodRationale		string
A value that identifies the rationale where electronic submission method is not to be allowed		
tender.submissionMethodDetails		string
Any detailed or further information on the submission method.		
tender.tenderPeriod		object
The period when this part of CP will be open for submission		
tender.procurementMethodModalities		string
The modalities of the procurement method		
tender.auctionPeriod		object
General period of all auctions scheduled under specific contracting process		
tender.electronicAuctions	array	object
See ElectronicAuctions		
tender.documents	array	object
All documents and attachments related to the tender, including any notices		
relatedProcesses	array	object

Objects described information about related funding (FSs), EI and other related sub-processes

`Parties` array object

Object described information about Entities involved into this part of CP

Command model

Title	Description	Obligation
`planning.rationale`		optional
`planning.budget.budgetBreakdown.id`		required
`planning.budget.budgetBreakdown.amount`		required
`tender.title`		required
`tender.description`		required
`tender.procurementMethodAdditionalInfo`		optional
`tender.submissionLanguages`	Has to be specified	optional
`tender.legalBasis`		required
`tender.procuringEntity`		required
`tender.tenderPeriod.startDate`		required
`tender.documents`		optional
`tender.procurementMethodRationale`		optional
`tender.procurementMethodModalities`		optional
`tender.electronicAuctions.id`		optional
`tender.electronicAuctions.details.relatedLot`		optional
`tender.electronicAuctions.details.electronicAuctionModalities`		optional
`tender.lots`		required
Element attributes		
`tender.lots.id`		required
`tender.lots.title`		required
`tender.lots.description`		optional
`tender.lots.value`		required
Validation rules: • *tender.lot.value.currency in all added lots has to be equal to budget.budgetBreakdown.amount.currency* • *the cumulative sum of all added tender.lot.value.amount has to be equal or less than sum of all added budget.budgetBreakdown.amount.amount*		
`tender.lots.contractPeriod`		required
Validation rules: • *tender.lot.contractPeriod.startDate has to be earlier than Tender.lot.contractPeriod.endDate in one lot object & tender.lot.contractPeriod.startDate has to be later than tender.tenderPeriod.endDate* • *earliest value among all added tender.lot.contractPeriod.startDate has to be earlier than values of all added planning.budget.budgetBreakdown.period.endDate* • *the latest value among all added tender.lot.contractPeriod.endDate has to be later than values of all added planning.budget.budgetBreakdown.period.startDate*		
`tender.lots.placeOfPerformance`		required
`tender.items`		required
Element attributes		
`tender.items.id`		required
`tender.items.description`		optional
`tender.items.relatedLots`		required
Validation rules: *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*		
`tender.item.classification`		required
Validation rules: *All added tender.Item.classification.id in all items coincide by first 3 figures of value of tender.classification.id in EI*		
`tender.items.additionalClassifications`		optional

tender.items.quantity	required
tender.items.unit	required
tender.enquiryPeriod.endDate	required
purposeOfNotice.isACallForCompetition	required
tender.lotGroups.optionToCombine	optional
tender.requiresElectronicCatalogue	optional

Update model

Title	Description
planning.rationale	Can be amended as free text
tender.title	Can be amended as free text
tender.description	Can be amended as free text
tender.enquiryPeriod.endDate	Deadline for clarification can be postponed. Submission period deadline will be rescheduled according to the rules relevant to the procedure under used legal basis
tender.tenderPeriod.endDate	Deadline for submission can be postponed considering minimum eligible duration according to used procedure
tender.documents	Set of document can be updated with new elements or previously published documents can be updated with new versions
Validation rules: *All the document that exist in initially created CN must be present in the payload of update request*	
tender.procurementMethodRationale	Can be amended as free text
tender.lots	Set of lots can be updated with new elements by splitting existing one to two or more separate lots with the appropriate allocation of the initial budget for splitting lot. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*
Note that: *In case of exclusion one or more lots from update-request, all related items will stay in release package of CN*	
Fields that can be amended	
tender.lots.title	Can be amended as free text
tender.lots.description	Can be amended as free text
tender.lots.contractPeriod	Period over which the contract is estimated or required to be active for this lot
Note that: • *any updates of lot.contractPeriod will have an influence to common tender.contractPeriod* • *any updates of lot.contractPeriod will be validated according to used FS and their planning.budget.budgetPeriod*	
tender.lots.placeOfPerformance	Can be amended as an object
tender.items	Set of items can not be changed but initially existing items can be amended with new values of following fields. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*
Note that: *In case of exclusion of one or more lots from update-request, all related items will stay in release package of CN with no connection.*	
Fields that can be amended	
tender.items.description	Can be amended as free text
tender.items.relatedLots	Relation with one of initial or added lots can be changed
Validation rules: *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*	

4.2.1.4 Tendering

Publication of CN means that CA has an intention to conduct a tender and start to collect the offers / intention of interests / quotations. Depending on procurement method, the CN may be used either to award a contract (single-stage competition) or to pre-select participants for restricted competition (multi-stage competition). In this respect CN might be different from data-models' perspective:

Pre-selection

This is a stage under entire contracting process during which interested parties can intend their interest to participate in future procedure by submitting a short statement of information about the organization according to instructions or a questionnaire. This form the basis of a qualification submission that such parties must make to demonstrate their compliance to eligibility criteria and minimal technical requirements. CA selects several candidates to be invited to next stage. This stage relates specifically to the selection of competent participants, prior to the issue of the invitations to tender with a main purpose to select those potential suppliers, whose qualifications and experience would minimise the risk of non-performance.

'PS' data-model

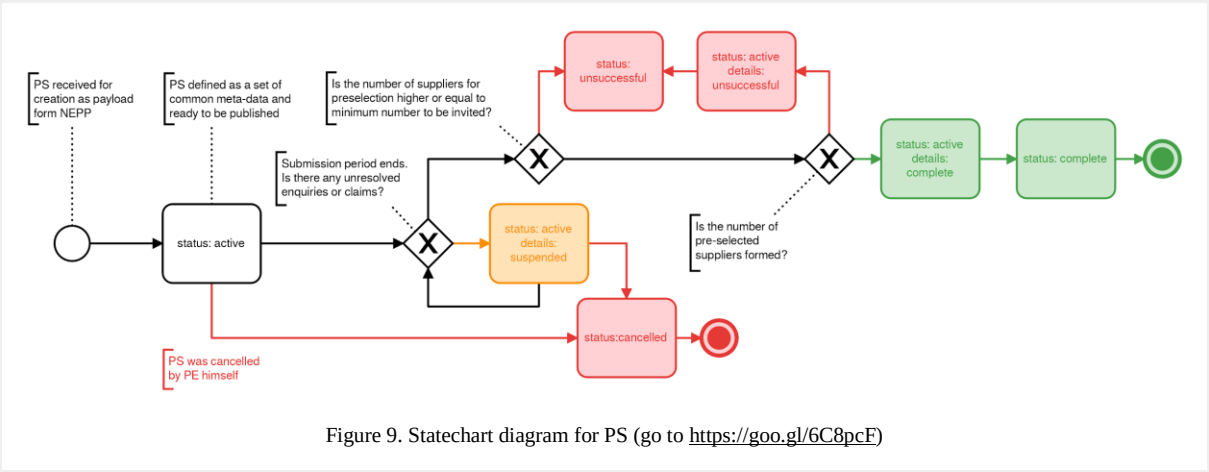


Figure 9. Statechart diagram for PS (go to https://goo.gl/6C8pcF)

Statechart statuses

Request of Interests Notice statuses

The *status f*ield is used to indicate the current status of a tender process.

Title	Description
`Active`	A tender process is currently taking place
`Cancelled`	The tender process has been cancelled.
`Unsuccessful`	The tender process was unsuccessful.
`Complete`	The tender process is complete.

Request of Interests Notice status details

The *statusDetails* field is used to indicate the current state of a tender process.

Title	Description
`Cancelled`	The contracting process is sent for cancellation but not yet cancelled
`Suspended`	The contracting process suspended due to business logic applied

Query model

Title, description		Type
`ocid`		string
A globally unique identifier for this part of Contracting Process		
`id`		string
An identifier for this particular release of information for part of CP		
`date`		date-time
The date this information is released		
`tag`		string
A value that identifies the nature of the release being made		
`initiationType`		string
String specifying the type of initiation process used for this part of CP		
`purposeOfNotice.isACallForCompetition`		boolean
A True/False field to indicate whether this notice is a call for competition		
`tender.id`		string
An identifier for this part of CP		
`tender.title`		string
Title for this part of CP		
`tender.description`		string
Description for this part of CP		
`preQualification.id`		string
An identifier for this pre-qualification process.		
`preQualification.title`		string
Pre-qualification title		
`preQualification.description`		string
Pre-qualification description		
`preQualification.status`		string
The current status of the pre-qualification		
`preQualification.statusDetails`		string
The current status details of the pre-qualification		
`preQualification.period`		object
The period when the pre-qualification is open for submissions.		
`preQualification.enquiryPeriod`		object
The period during which enquiries regarding the pre-qualification may be made and answered		
`preQualification.procuringEntity`		object
The entity managing the procurement, which may be different from the buyer who is paying / using the items being procured		
`preQualification.documents`	array	object
All documents and attachments related to the pre-qualification, including any notices		
`preQualification.milestones`	array	object
A list of milestones associated with the pre-qualification		
`tender.status`		string
The current status of this part of CP based on the table of statuses		
`tender.statusDetails`		string
The current status of this part of CP based on the table of statuses		

`tender.lotGroups.optionToCombine`		boolean
True/False value indicates the CA reserves the right to combine the lots in this group when awarding a contract		
`tender.lots`	array	object
A tender process is divided into lots.		
`tender.items`	array	object
The goods and services to be purchased		
`tender.requiresElectronicCatalogue`		boolean
True/False value indicates whether bids must include an electronic catalogue.		
`tender.procurementMethodModalities`		string
The modalities of the procurement method		
`tender.documents`	array	object
All documents and attachments related to the tender, including any notices		
`relatedProcesses`	array	object
Objects described information about related funding (FSs), EI and other related sub-processes		
`parties`	array	object
Object described information about Entities involved into this part of CP		

Command model

Title	Description	Obligation
`planning.rationale`		optional
`planning.budget.budgetBreakdown.id`		required
`planning.budget.budgetBreakdown.amount`		required
`tender.title`		required
`tender.description`		required
`tender.procurementMethodAdditionalInfo`		optional
`tender.legalBasis`		required
`tender.procuringEntity`		required
`tender.documents`		optional
`tender.procurementMethodRationale`		optional
`tender.procurementMethodModalities`		optional
`tender.lots`		required
Element attributes		
`tender.lots.id`		required
`tender.lots.title`		required
`tender.lots.description`		optional
`tender.lots.value`		required
Validation rules: • *tender.lot.value.currency in all added lots has to be equal to budget.budgetBreakdown.amount.currency* • *the cumulative sum of all added tender.lot.value.amount has to be equal or less than sum of all added budget.budgetBreakdown.amount.amount*		
`tender.lots.contractPeriod`		required
Validation rules: • *tender.lot.contractPeriod.startDate has to be earlier than Tender.lot.contractPeriod.endDate in one lot object & tender.lot.contractPeriod.startDate has to be later than tender.tenderPeriod.endDate* • *earliest value among all added tender.lot.contractPeriod.startDate has to be earlier than values of all added planning.budget.budgetBreakdown.period.endDate* • *the latest value among all added tender.lot.contractPeriod.endDate has to be later than values of all added planning.budget.budgetBreakdown.period.startDate*		

tender.lots.placeOfPerformance	required
tender.items	required
Element attributes	
tender.items.id	required
tender.items.description	optional
tender.items.relatedLots	required
Validation rules: *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*	
tender.item.classification	required
Validation rules: *id in all items coincide by first 3 figures of value of tender.classification.id in EI*	
tender.items.additionalClassifications	optional
tender.items.quantity	required
tender.items.unit	required
purposeOfNotice.isACallForCompetition	required
tender.lotGroups.optionToCombine	optional
tender.requiresElectronicCatalogue	optional
preQualification.title	optional
preQualification.description	optional
preQualification.period.endDate	mandatory
preQualification.procuringEntity	optional
preQualification.documents	optional

Update model

Title	Description
planning.rationale	Can be amended as free text
tender.title	Can be amended as free text
tender.description	Can be amended as free text
tender.documents	Set of document can be updated with new updated with new versions
Validation rules: *All the document that exist in initially created CN must be present in the payload of update request*	
tender.lots	Set of lots can be updated with new elements by splitting existing one to two or more separate lots with the appropriate allocation of the initial budget for splitting lot. Initially published lots can be switched to *cancelled by excluding of such lot from update request.*
Note that: *In case of exclusion one or more lots from update-request, all related items will stay in release package of CN*	
Fields that can be amended	
tender.lots.title	Can be amended as free text
tender.lots.description	Can be amended as free text
tender.lots.contractPeriod	Period over which the contract is estimated or required to be active for this lot
Note that: *any updates of lot.contractPeriod will have an influence to common tender.contractPeriod and any updates of lot.contractPeriod will be validated according to used FS and their planning.budget.budgetPeriod*	
tender.lots.placeOfPerformance	Can be amended as an object
tender.items	Set of items can not be changed but initially existing items can be amended with new values of following fields. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*

Note that: *In case of exclusion of one or more lots from update-request, all related items will stay in release package of CN with no connection.*

Fields that can be amended	
`tender.items.description`	Can be amended as free text
`tender.items.relatedLots`	Relation with one of initial or added lots can be changed
Validation rules: *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*	
`preQualification.title`	
`preQualification.description`	
`preQualification.period.endDate`	
`preQualification.documents`	

Evaluation

During 'Evaluation' stage all invited (pre-selected) parties submitting a financial offer according to instructions or a questionnaire. CA selects several candidates to be invited to next stage. Financial evaluation of the proposals involves consideration the offers against financial and commercial requirements stipulated in the tender documentation and in accordance to awarding criteria used.

'EV' data-model

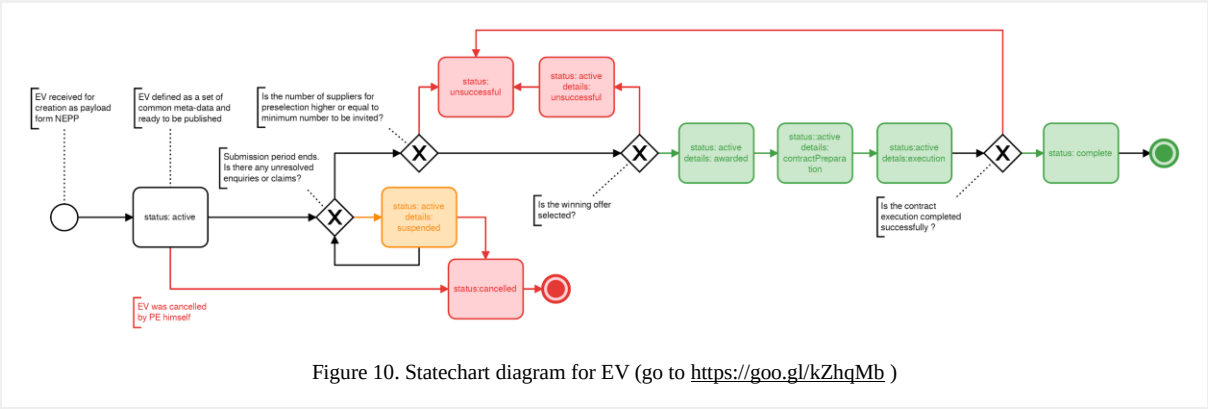


Figure 10. Statechart diagram for EV (go to https://goo.gl/kZhqMb)

Statechart statuses

Request of Interests Notice statuses

The *status f*ield is used to indicate the current status of a tender process.

Title	Description
`active`	A tender process is currently taking place
`cancelled`	The tender process has been cancelled.
`unsuccessful`	The tender process was unsuccessful.
`complete`	The tender process is complete.

Request of Interests Notice status details

The *statusDetails* field is used to indicate the current state of a tender process.

Title	Description
`cancelled`	The contracting process is sent for cancellation but not yet cancelled
`suspended`	The contracting process suspended due to business logic applied
`awarded`	The specific tenderer is awarded within current phase of the contracting process
`contractPrearation`	Contract preparation actions starts under this phase of the contracting process
`execution`	Contract execution runs under this phase of the contracting process

Query model

Title, description		Type
ocid		string
A globally unique identifier for this part of Contracting Process		
id		string
An identifier for this particular release of information for part of CP		
date		date-time
The date this information is released		
tag		string
A value that identifies the nature of the release being made		
initiationType		string
String specifying the type of initiation process used for this part of CP		
purposeOfNotice.isACallForCompetition		boolean
A True/False field to indicate whether this notice is a call for competition		
tender.id		string
An identifier for this part of CP		
tender.title		string
Title for this part of CP		
tender.description		string
Description for this part of CP		
tender.status		string
The current status of this part of CP based on the table of statuses		
tender.statusDetails		string
The current status of this part of CP based on the table of statuses		
tender.lotGroups.optionToCombine		boolean
True/False value indicates the CA reserves the right to combine the lots in this group when awarding a contract		
tender.lots	array	object
A tender process is divided into lots.		
tender.items	array	object
The goods and services to be purchased		
tender.requiresElectronicCatalogue		boolean
True/False value indicates whether bids must include an electronic catalogue.		
tender.submissionMethod		string
Specify the method by which bids must be submitted		
tender.submissionMethodRationale		string
A value that identifies the rationale where electronic submission method is not to be allowed		
tender.submissionMethodDetails		string
Any detailed or further information on the submission method.		
tender.tenderPeriod		object
The period when this part of CP will be open for submission		
tender.procurementMethodModalities		string
The modalities of the procurement method		
tender.auctionPeriod		object
General period of all auctions scheduled under specific contracting process		
tender.electronicAuctions	array	object
See ElectronicAuctions		
tender.documents	array	object
All documents and attachments related to the tender, including any notices		
relatedProcesses	array	object
Objects described information about related funding (FSs), EI and other related sub-processes		
parties	array	object

Object described information about Entities involved into this part of CP

Command model

Title	Description	Obligation
`planning.rationale`		optional
`planning.budget.budgetBreakdown.id`		required
`planning.budget.budgetBreakdown.amount`		required
`tender.title`		required
`tender.description`		required
`tender.procurementMethodAdditionalInfo`		optional
`tender.submissionLanguages`		optional
`tender.legalBasis`		required
`tender.procuringEntity`		required
`tender.tenderPeriod.startDate`		required
`tender.documents`		optional
`tender.procurementMethodRationale`		optional
`tender.procurementMethodModalities`		optional
`tender.electronicAuctions.id`		optional
`tender.electronicAuctions.details.relatedLot`		optional
`tender.electronicAuctions.details.electronicAuctionModalities`		optional
`tender.lots`		required
Element attributes		
`tender.lots.id`		required
`tender.lots.title`		required
`tender.lots.description`		optional
`tender.lots.value`		required
Validation rules:		
● *tender.lot.value.currency in all added lots has to be equal to budget.budgetBreakdown.amount.currency*		
● *the cumulative sum of all added tender.lot.value.amount has to be equal or less than sum of all added budget.budgetBreakdown.amount.amount*		
`tender.lots.contractPeriod`		required
Validation rules:		
● *tender.lot.contractPeriod.startDate has to be earlier than Tender.lot.contractPeriod.endDate in one lot object & tender.lot.contractPeriod.startDate has to be later than tender.tenderPeriod.endDate*		
● *earliest value among all added tender.lot.contractPeriod.startDate has to be earlier than values of all added planning.budget.budgetBreakdown.period.endDate*		
● *the latest value among all added tender.lot.contractPeriod.endDate has to be later than values of all added planning.budget.budgetBreakdown.period.startDate*		
`tender.lots.placeOfPerformance`		required
`tender.items`		required
Element attributes		
`tender.items.id`		required
`tender.items.description`		optional

`tender.items.relatedLots`	required
Validation rules: *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*	
`tender.item.classification`	required
Validation rules: *All added tender.Item.classification.id in all items coincide by first 3 figures of value of tender.classification.id in EI*	
`tender.items.additionalClassifications`	optional
`tender.items.quantity`	required
`tender.items.unit`	required
`tender.enquiryPeriod.endDate`	required
`purposeOfNotice.isACallForCompetition`	required
`tender.lotGroups.optionToCombine`	optional
`tender.requiresElectronicCatalogue`	optional
`preQualification.title`	optional
`preQualification.description`	optional
`preQualification.period.endDate`	mandatory
`preQualification.procuringEntity`	optional
`preQualification.documents`	optional

Update model

Title	Description
`planning.rationale`	Can be amended as free text
`tender.title`	Can be amended as free text
`tender.description`	Can be amended as free text
`tender.documents`	Set of document can be updated with new elements or previously published documents can be updated with new versions
Validation rules: *All the document that exist in initially created CN must be present in the payload of update request*	
`tender.procurementMethodRationale`	Can be amended as free text
`tender.lots`	Set of lots can be updated with new elements by splitting existing one to two or more separate lots with the appropriate allocation of the initial budget for splitting lot. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*
Note that: *In case of exclusion one or more lots from update-request, all related items will stay in release package of CN*	
Fields that can be amended	
`tender.lots.title`	Can be amended as free text
`tender.lots.description`	Can be amended as free text
`tender.lots.contractPeriod`	Period over which the contract is estimated or required to be active for this lot
Note that: • *any updates of lot.contractPeriod will have an influence to common tender.contractPeriod* • *any updates of lot.contractPeriod will be validated according to used FS and their planning.budget.budgetPeriod*	
`tender.lots.placeOfPerformance`	Can be amended as an object
`tender.items`	Set of items can not be changed but initially existing items can be amended with new values of following fields. Initially published lots can be switched to *status:cancelled by excluding of such lot from update request.*

Note that: *In case of exclusion of one or more lots from update-request, all related items will stay in release package of CN with no connection.*

Fields that can be amended	
`tender.items.description`	Can be amended as free text
`tender.items.relatedLots`	Relation with one of initial or added lots can be changed

Validation rules: *Values of tender.items.relatedLot for all Items have to be matched with all delivered lots by tender.lots.id*

4.2.1.5Awarding

Contract Award Notice

Contract Award Notice is a public announcement of the outcome of a public procurement exercise. CAN indicates a results of the tendering and awarding process for both positive and negative cases.

'CAN' data-model

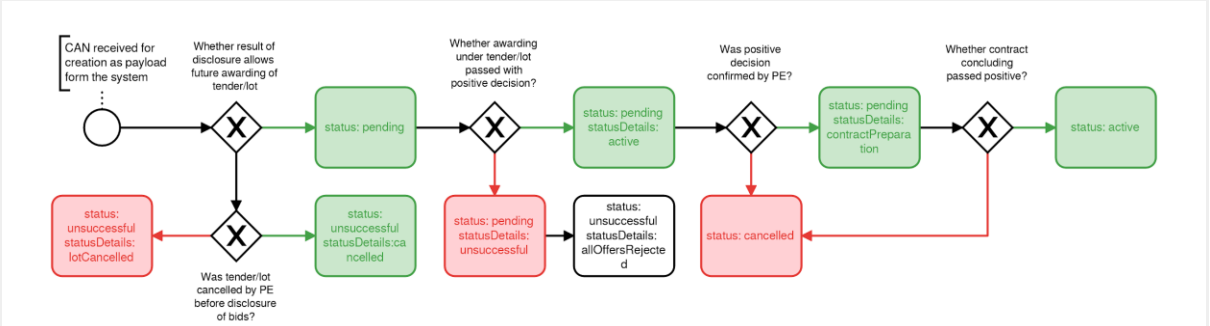


Figure 12. Statechart diagram for CAN (go to https://goo.gl/mxFYZz)

Statechart statuses

Request of Interests Notice statuses

The *status f*ield is used to indicate the current status of a Contract Award Notice.

Title	Description
`active`	The CAN is published and indicates positive decision regarding awarding in the lot
`cancelled`	The CAN is cancelled by CA
`unsuccessful`	The CAN is published and indicates negative decision regarding awarding in the lot

Request of Interests Notice status details

The *statusDetails* field is used to indicate the current state of Contract Award Notice.

Title	Description
`lotCancelled`	The reason why the award is negative is because this lot was cancelled by CA
`unsuccessful`	The CAN is going to be published as negative once stand-still period expired
`allOffersRejected`	The reason why the award is negative is because all the offers received are rejected
`active`	The CAN is going to be published as positive once stand-still period expired
`contractPreparation`	The contract preparation under this CAN is in progress

`noOffersReceived`	The reason why the award is negative is because there was no offers received

Query model

Query-model of Contract Award Notice is based on definition of ‘Contract’ object but used with cutted scheme

Title, description	Type		Obligation
id The identifier for this CAN		string	auto-generated
awardID The identifier of award against which this CAN is being issued		string	auto-generated
date Date of publication of CAN		date	auto-generated
documents	array	object	auto-generated
status The current status of the CAN		string	auto-generated
statusDetails The details about current status of the CAN		string	auto-generated
relatedProcesses If there is already AC generated against this CAN the identifier of such AC will be provided here	array	object	auto-generated
amendments A CAN amendment is an object indicates that this CAN was sent for cancellation	array	object	auto-generated

Reference to Awarded Contract

Upon the completion of complaining process system will allow to generate a set of separate Release Packages describing a scope and details of each awarded contract related with each ‘award:active’. Such relation describes within ‘RelatedProcesses’ object as an attribute of specific item of ‘contracts’ array.

```
{
  "contracts": [
    {
      "id": "",
      "awardId": "",
      "status": "",
      "statusDetails": "",
      "date": "",
      "documents": [],
      "amendments":                                                    [],
      "relatedProcesses": [
        {
          "$ref": "#/definitions/relatedProcess"
        }
      ]
    }
  ]
}
```

Update model

Once CAN is published and before CAN is confirmed Contracting Authority able to update such CAN with any additionally required document (e.g. report on procedure or signed decision of tender committee)

4.2.1.6 Contracting

The contract section is used to provide details of contracts that have been entered into. Every contract needs to have a related award, linked via the awardID property. This is because supplier information is contained within the 'award'. The framework contract details below help illustrate the reasons for this.

Awarded Contract

Awarded Contract is a result of the tendering procedure - concluded contract based on previously published Contract Award Notice.

'AC' data-model

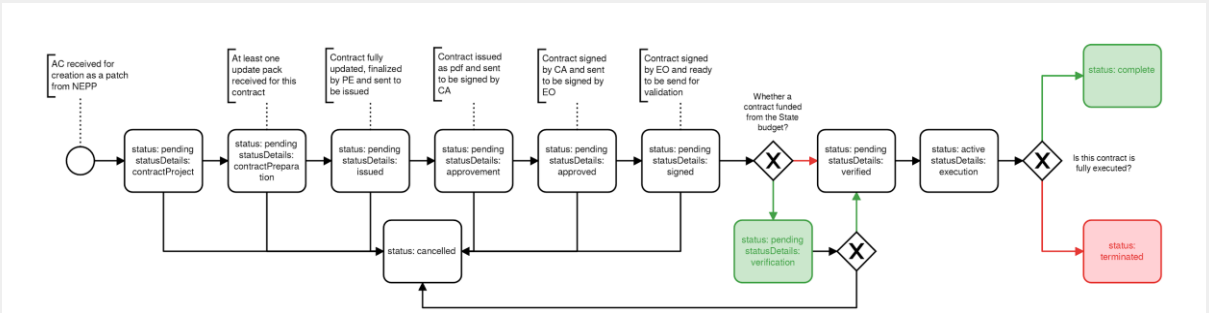


Figure 13. Statechart diagram for AC (go to https://goo.gl/SHiQS3)

Statechart statuses

Request of Interests Notice statuses

The *status f*ield is used to indicate the current status of a Contract

Title	Description
`pending`	This contract has been proposed, but is not yet in force.
`active`	This contract has been signed by all the parties, and is now legally in force.
`cancelled`	This contract has been cancelled prior to being signed.
`terminated`	This contract was signed and in force, and has now come to a close.

Request of Interests Notice status details

The *statusDetails* field is used to indicate the current state of Contract

Title	Description
`contractProject`	This contract has been issued as a draft and needs to be prepared by CA
`issued`	This contract has been fully prepared by CA and ready for approvement by CA
`approvement`	This contract is under approvement by CA
`approved`	This contract has been signed by CA and moved for signing by EO
`signed`	This contract has been signed by both, CA and EO
`verification`	This contract has been sent for verification to the State Treasury

`verified`	This contract has been verified by State Treasury
`clarification`	This contract has been returned by State Treasury with clarification request
`rejected`	This contract has been rejected by State Treasury
`cancellation`	This contract has been set for cancellation (termination before signed by Supplier)
`termination`	This contract has been set for termination
`complete`	This contract has been terminated as completed
`unsuccessful`	This contract has been terminated as unsuccessful

Query model

'Awarded Contract' is a model that describes a contract obtained as a result of a competitive procurement procedure or the registration of a direct contract.

Title, description	Type	Obligation
`ocid` A globally unique identifier for this part of contract	string	
`id` An identifier for this particular release of information for the contract	string	
`date` The date this information is released	date-time	
`tag` A value that identifies the nature of the release being made	string	
`initiationType` String specifying the type of initiation process used for this part of CP	string	
`tender` The activities undertaken in order to enter into this contract	object	
Object's specifics for AC		
The data for this object is collected from CP and CN. The data here is used to identify main params of this Awarded Contract without reading of other parts of the Record Package of parent Contracting Process		
`tender.id` An identifier taken from CP	string	
`tender.classification` The primary classification for the tender taken from CP	object	
`tender.procurementMethod` Tendering method code taken from CP	string	
`tender.procurementMethodDetails` Additional detail on the procurement method taken from CN	string	
`tender.mainProcurementCategory` The primary category describing the object of this contracting process taken from CP	string	
`tender.lots` Required details of each lot taken from CN See Lot	array	object
`awards`	array	object

Information from the evaluation phase of the contracting process.
See Award

Object's specifics for AC

Marged information of awards included into this concluded contract. There may be information collected from different awards published on evaluation phase - in case if selected winner for both (all) awards is the same legal entity

award.id		string
Separate identifier for award under Awarded Contract		
award.date		string
Date of		
award.description		string
In case of merging of two or more awards from evaluation phase under single Awarded Contract - justification of such merging		
award.value		object
Total amount and its details for this specific award as a scope of Awarded Contract. **Important:** on this (contracting) stage previously (evaluation) used as specification of amount of selected financial offer without VAT *award.value.amount* goes to *award.value.amountNet. And award.value.amount here specifies the total amount of price for all deliverables under this awarded contract with VAT.*		
award.suppliers	array	object
The suppliers awarded this award.		
award.items	array	object
The subject collected under this specific awarded contract, broken into lines. In case of merging of two or more awards from evaluation phase, all items related to lots under merged awards described here as a single list		
award.relatedLots	array	string
Set of identifiers of lots described in 'tender' object above related to this award		
contracts	array	object
The set of information of concluded contract		
relatedProcesses	array	object
Objects described information about related funding (FSs), EI and other related sub-processes		
parties	array	object
Object described information about Entities involved into this concluded contract. Information here has be specified in order to recommendations of this documentation		
planning		object
Planning for this contract		

Object's specifics for AC

Marged information of awards included into this concluded contract. There may be information collected from different awards published on evaluation phase - in case if selected winner for both (all) awards is the same legal entity

planning.implementation.transactions	array	object
Set of financial transactions planned under this specific concluded contract. Each planned transaction can be related to on specific deliverable or milestone achieved by 'relatedContractMilestone'		
planning.budget.budgetSource	array	object
List of the Funding Sources used under this awarded contract. It may be cutted or even different comparing with one in 'cnParent' of parent Contracting Process		
planning.budget.budgetAllocation	array	object
Information about allocation of used fundings from three main angles: what, when, what for: each Funding Source described above has to be specified with period of use and related one of items from list of described under this 'award'. In case if one particular funding source is used for two or more items, If different items are funded from same funding source, these should be split into separate budgetAllocation blocks in parent array.		

Command models

To create AC standard data-model should be used for preparation of POST-request for BPE. Such POST-request must be based on definition of Contract and include only required and optional data-fields.

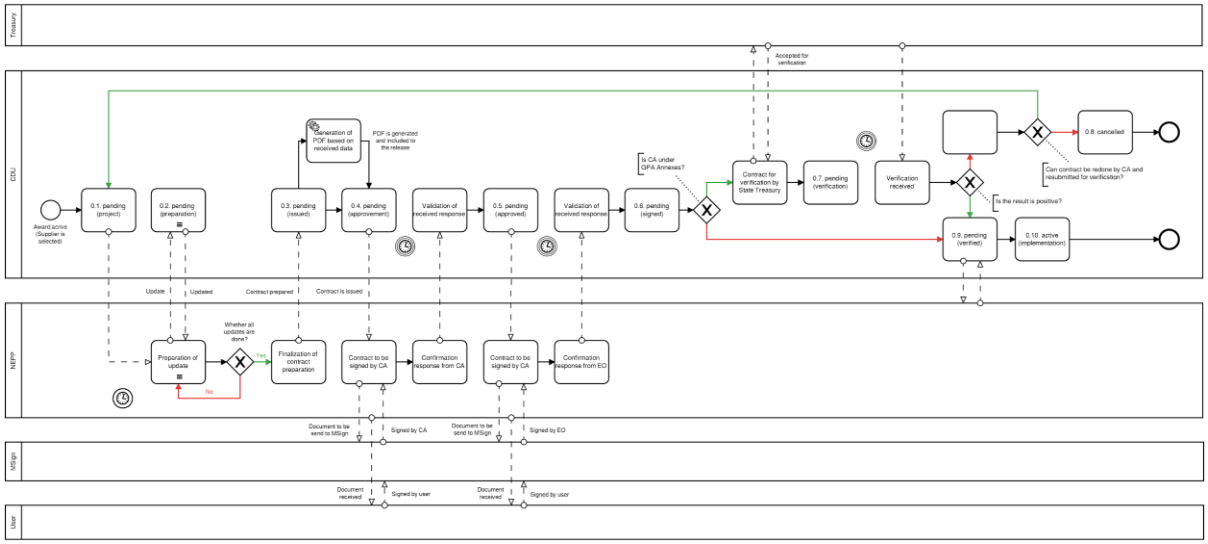


Figure 13.1. General contract preparation, signing, verification and activation process (go to https://goo.gl/jEkWtS)

Create a draft of awarded contract

Once the review period has expired, Awarded Contract against one or several CANs can be created by indication of the CAN(s) using the following scheme of the command mode:

Title, description	Obligation
`contract.id` The identifier for CAN	required

Update draft of awarded contract and its preparation for issuing

Since '*contractProject*' is generated by system, updates can be prepared according to following scheme:

Title	Description
`planning.implementation.transactions`	
`planning.budget.description`	
`planning.budget.budgetSource`	
`planning.budget.budgetAllocation`	
`contract.extendsContractID`	
`contract.title`	
`contract.description`	
`contract.period`	
`contract.agreedMetrics[*].observations[*].measure`	
`contract.documents`	

contract.milestones
awards[*].id
awards[*].suppliers[*].id
awards[*].suppliers[*].additionalIdentifiers
awards[*].suppliers[*].persones
awards[*].suppliers[*].details.typeOfSupplier
awards[*].suppliers[*].details.mainEconomicActivity
awards[*].suppliers[*].details.scale
awards[*].suppliers[*].details.permits
awards[*].suppliers[*].details.bankAccounts
awards[*].suppliers[*].details.legalForm
awards[*].documents
awards[*].value.amount
awards[*].value.currency
awards[*].value.amountNet
awards[*].value.valueAddedTaxincluded
items[*].id
items[*].quantity
items[*].unit.value.amount
items[*].unit.value.currency
items[*].unit.value.amountNet
items[*].unit.value.valueAddedTaxIncluded
items[*].deliveryAddress
buyer.id
buyer.additionalIdentifiers
buyer.persones
buyer.details.typeOfBuyer
buyer.details.mainGeneralActivity
buyer.details.mainSectoralActivity
buyer.details.bankAccounts
buyer.details.legalForm

Termination of issued Contract

CA can terminate a contract anytime starting from the moment of generation. For this next scheme has to be used:

Title	Description
amendment.rationale	
amendment.description	
amendment.documents	

Signing the Contract

Since contract is issued and PDF version is generated and added to the contract.documents by the system, contract will go to '*statusDetails:approvement*' and will be ready to signed by Contracting Authority. To confirm that contract signed by CA, next scheme has to be used:

Title	Description

```
confirmationResponses[].value.id
```

```
confirmationResponses[].value.date
```

```
confirmationResponses[].value.relatedPerson.id
```

```
confirmationResponses[].value.verification.type
```

```
confirmationResponses[].value.verification.value
```

4.2.1.7User Actions

Clarification

During ‘enquiryPeriod’ all data describing this period go to public model in real-time. All Entities who participates as ‘enquirer’ will be indicated in initial ‘parties’ right after such information became a public

Enquiries

Extension of initial query-model

All submitted questions (enquiries) describes as an array ‘enquiries’ in the definition of ‘tender’.

```
{
  "tender":                                                    {
    "enquiries":                                               [
      {"$ref":                                  "#/definitions/Enquiry"}
    ]
}
```

Also ‘hasEnquiries’ in cnParent will be switched in ‘true’ automatically in case of existing enquiries.

```
{
  "tender":                                                    {
    "hasEnquiries":                                          "true"
}
```

Command Model

To create new question standard POST data model should be used for preparation of POST-request.

Schema

enquiry.author Question author See Organization	object	required
enquiry.title The subject line of the question	string	required
enquiry.description The body of the question	object	required
enquiry.relatedLot If this question relates to a specific lot, this field contains the lot identifier.	string	required

Example

```
{
  "enquiry":                                                   {
    "author":                                                  {
      "$ref":                                "#/definitions/Organization"
    },
    "title":                                                   "",
    "description":                                             "",
    "relatedLot":                                              ""
  }
```

```
}
```

See how to submit question in Tutorial

Answers

During ‘enquiryPeriod” CA is able to submit an answer to question received.

Extension of initial ‘Enquiry’ data-model

```
{
  "enquiries":                                                    [
    {
      "answer":"",
      "dateAnswered":""
    }
  ]
}
```

Command Model

To submit an answer to a specific question standard data-model should be used for preparation of POST-request

Schema

answer	string	required
Free text of answer		

Example

```
{
  "enquiry":                                                      {
    "answer":                                  "Lorem                                  Ipsum"
  }
}
```

See how to submit question in Tutorial

Future steps

Submission

‘bids‘ section will be included to initial query-model of the stage and will include all ‘bids‘ that were disclosed Initial status of such ‘bids’ will be ‘pending’. All Entities who submitted ‘bids’ will be indicated in initial ‘parties’ as ‘tenderer’. After ‘tenderPeriod‘ all data describes this period goes to public.

Submitted and disclosed bids

Extension of initial ‘Process Stage’ query-model

```
{
  "bids":                                                                    {
    "details":                                                               [
      {"$ref":                                                  "#/definitions/Bid"}
    ]
  }
}
```

Command Model

To submit a bid standard data-model should be used for preparation of POST-request for BPE.

Schema

bid.tenderers The party, or parties, responsible for this bid. See Organization	array	object	required
bid.value The total value of the bid See Value		object	required
bid.documents All documents and attachments related to the bid See Document	array	object	optional
bid.requirementResponses The responses related to the requirements	array	object	optional
bid.relatedLots Provide the lot identifier here.		string	required

Example

```
{
  "bid":                                                                     {
    "tenderers":                                                            [
      {
        "$ref": "#/definitions/Organization"
      }
    ],
    "value":                                                                {
      "$ref":                                                 "#/definitions/Value"
    },
    "documents":                                                            [
      {
        "$ref": "#/definitions/Document"
      }
    ],
    "requirementResponses":[{
        "$ref": "#/definitions/requirementResponses"
      }
    ],
```

```
    "relatedLots":                                                    [
      ""
    ]
  }
}
```

Requirement responses

There is a number of scenarios in which structured information on requirements and other procurement conditions is included in the Contract Notice (see CN publication future steps). The requirements extension used for communicating criteria on the CN level also includes structures for collecting relevant requirement responses related to described requirements. Thus, command model of '*bid*' will be extended with *'requirementResponses*' array cover all the '*criteria*' specified in the Contract Notice

See how to submit a bid in Tutorial

Update of submitted bid

Anytime during the contracting process before the deadline of submission period, Economic Operator is able to update previously submitted bid. To update existing entity of bid following model should be used for preparation of POST-request for BPE:

Command model

Schema

bid.tenderers	Must be included as it is in initial bid
bid.value.amount	Can be amended as free figures
bid.documents	Set of document can be updated with new elements or previously published documents can be updated with new versions
bid.relatedLot	Must be specified as it is for initial bid
bid.requirementResponses	Set of responses provided for the requirements

Withdrawal of submitted bid

Anytime during the contracting process before the deadline of submission period, Economic Operator is able to withdraw his previously submitted bid. To withdraw existing entity of bid POST-request according to BPE rules should be used without any extensions for the data model of *bid*

Update of bid selected as a winning

Once Contracting Authority has selected particular bid as a winning (relevant *award:status:active*) and before the Awarded Contract is issued, Economic Operator is able to update his previously submitted *bid* with any required *documents*, requested by Contracting Authority (e.g. confirmation of self-declaration, actualized or detailed commercial offer, etc). To update existing entity of *bid* with set of new (or updated) *documents* following model should be used for preparation of POST-request for BPE:

Command model

Schema

bid.documents	Set of document can be updated with new elements or previously published documents can be updated with new versions

Direct Awarding

'awards' section will be included to initial query-model of the stage and will include all 'awards' that were introduced, Initial status of such 'awards' will be 'pending'. All Entities who were sent as 'suppliers' in 'awards' will be indicated in initial 'parties' as 'suppliers'.

Introduced awards

Extension of initial 'Process Stage' query-model

```
{
  "awards":                                                                    [
    {
      "$ref": "#/definitions/Award"
    }
  ]
}
```

Command Model

To introduce an award standard data-model should be used for preparation of POST-request for BPE.

Schema

award.suppliers The party, or parties, responsible for this award. See Organization	array	object	required
award.value The total value of the award See Value		object	required
award.description Description of an award		string	optional
award.relatedLot Provide the lot identifier here.		string	required

Example

```
{
  "award": {
    "description"                                  :                                  "",
    "suppliers":                                                                      [
      {
        "$ref": "#/definitions/Organization"
      }
    ],
    "value":                                                                          {
      "$ref":                                                          "#/definitions/Value"
    },
    "relatedLots":                                                                    [
      ""
    ]
  }
}
```

Electronic reverse auction

If electronic auction included into specific contracting process, initial query-model of CN will be extended with the following:

On publication level

Subsection ‘procurementMethodModalities’ and ‘electronicAuctions’ of ‘tender’ section will be included to initial query-model.

```
{
  "tender": {
    "auctionPeriod": {
      "startDate": ""
    },
    "procurementMethodModalities": [
      "electronicAuction"
    ],
    "electronicAuctions": {
      "details": [
        {
          "id": "",
          "relatedLot": "",
          "auctionPeriod": {
            "startDate": ""
          },
          "electronicAuctionModalities": [
            {
              "eligibleMinimumDifference": {
                "amount": "",
                "currency": ""
              }
            }
          ]
        }
      ]
    }
  }
}
```

On execution level

Unique link for each scheduled electronic auction will be available in Public Point. Depending on access-mode (with or without token) this link might work as:

- individual link of the participant, who accesses his personal page via this link and participates in the auction.
- public link to the auction which can be published anywhere for viewers, who use this link to observe.

```
{
  "tender": {
    "electronicAuctions": {
      "details": [
        {
          "electronicAuctionModalities": [
            {
              "url": ""
            }
          ]
        }
```

```
      ]
    }
  }
}
```

On result level

Once all established under single submission stage electronic auctions end additional data describe final details and statistics will be included into initial query-model:

```
{
  "tender": {
    "auctionPeriod": {
      "endDate": ""
    },
    "electronicAuctions": {
      "details": [
        {
          "auctionPeriod": {
            "endDate": ""
          },
          "electronicAuctionProgress": [
            {
              "id": "1",
              "period": {
                "startDate": "",
                "endDate": ""
              },
              "breakdown": [
                {
                  "relatedBid": "",
                  "status": "",
                  "dateMet": "",
                  "value": {
                    "amount": "",
                    "currency": ""
                  }
                }
              ]
            }
          ],
          "electronicAuctionResult": [
            {
              "relatedBid": "",
              "value": {
                "amount": "",
                "currency": ""
              }
            }
          ]
        }
      ],
      "statistics": [ ]
    }
  }
}
```

Information disclosure on tenderers occurs once the last auction in this process is completed

Evaluation and awarding

During ‘awardPeriod’ all data describing this period go to the public model in real-time. All Entities who participate in the submission stage as ‘tenderer’ and become a ‘supplier’ (bidder of bid collects ‘award’ with ‘statusDetails:pending’) will be indicated in initial ‘parties’ right after such information became public.

Extension of initial ‘Process Stage’ data-model

```
{
  "tender":                                                    {
    "awardPeriod":                                             {
      "$ref":                                    "#/definitions/Period"
    }
  },
  "awards":                                                    [
    {
      "$ref":                                     "#/definitions/Award"
    }
  ]
}
```

Command Model for Declaration of non conflict of interest *by evaluation panel*

To submit a declaration of non conflict of interest by each separate evaluation panel member a standard data-model to be used for the preparation of the POST-request for BPE. Such POST-request must be based on definition of ‘award’ and include only required and optional data-fields.

Schema

requirementResponses	array	object	required
The needed statusDetails according See RequirementResponse			

Example

```
{
  "requirementResponses":                                      [
    {
      "id":                                  "nonConflictOfInterest-1-1",
      "value":                                                 true,
      "requirement":                                              {
        "id":"nonConflictOfInterest-1-1"
      },
      "responder":  {}                    // specific evaluation panel member to be reflected as
    },                                 // PersonReference or complete Persone object (for replacements)
    {
      "id":                                  "nonConflictOfInterest-1-2",
      "value":                                                 true,
      "requirement":                                              {
        "id":"nonConflictOfInterest-1-2"
      },
      "responder":                                                {}
    }
  ]
}
```

Command Model for Award *update*

To submit a decision via publication of updates of generated ‘award’ standard data-model should be used for preparation of POST-request for BPE. Such POST-request must be based on definition of ‘award’ and include only required and optional data-fields

Schema

statusDetails The needed statusDetails according to Award Status Details codelist		string	required
description Free-text description of decision made		string	optional
documents All documents and attachments related to the bid See Document	array	object	optional

Example

```
{
  "award":                                                    {
    "statusDetails":                                         "",
    "description":                                           "",
    "documents":                                               [
      {
        "$ref":                                    "#/definitions/Document"
      }
    }
  ]
}
```

See how to submit question in Tutorial

Working with Contract Award Notices

Since CANs generated as a result of received evaluation protocol, CA is able to run a set of operations

CAN update

Once CAN is published and before CAN is confirmed Contracting Authority able to update such CAN with any additionally required document (e.g. report on procedure or signed decision of tender committee)

Command Model

Schema

documents.id Identifier of document in Document Service	string	required
documents.documentType Type of document specified according to the codelist	string	required
documents.title Free text title of the document	string	required
documents.description Free text description of the document	string	optional

Example

```
{
  "documents":[
    {
      "documentType":                                                    "",
      "id":                                                              "",
      "title":                                                           "",
      "description":                                                      ""
    }
  ]
}
```

CAN cancellation

In case of a limited list of reasons published Contract Award Notice can be cancelled by CA. Such cancellation can be executed within submitting of separate object - amendment, contains all the information needed to execute and publish such cancellation.

Extension of initial 'CAN' query-model

Once cancellation for CAN submitted, initial model of CAN will be extended:

```
{
  "tag":                                                          "awardCancellation",
  "contracts":                                                                    [
    {
      "amendments":                                                               [
        {
          "id":                                                                  "",
          "status":"active",
          "statusDetails":"cancellation",
          "date":                                                                "",
          "amendsReleaseID":                                                     "",
          "releaseID":                                                           "",
          "rationale":                                                           "",
          "description":                                                         "",
          "documents":                                                            []
        }
```

```
      ]
    }
  ]
}
```

Command Model

To submit an amendment for CAN cancellation standard command-model should be used.

Schema

amendment.rationale	string		required
Reason for cancellation (selected from a standardized closed value-list)			
amendment.description	string		optional
The additional information about cancellation			
amendment.documents	array	object	optional
All documents and attachments related to the cancellation See Document			

Example

```
{
  "contract":                                                    {
    "amendment":                                                 {
      "rationale":                                              "",
      "description":                                            "",
      "documents":                                               [
        {
          "documentType":                                       "",
          "id":                                                 "",
          "title":                                              "",
          "description":                                         ""
        }
      ]
    }
  }
}
```

Cancellations

Anytime during contracting process before the contract is established, Procuring Entity is able to cancel either separate lot or all the contracting process. The cancellation mechanism technically implemented in a different way for all tender and separate lots.

Lot cancellation

Depending on the period and state of contracting process, lot cancellation flow can be executed in different ways:

During clarification period

Cancellation of the lot is executed within standart update of Contract Notice: identifier of lot or lots to be excluded needs to be absent in update-payload based on cn command-model.

Future steps

In a next release it is considered to use the same approach as for tender cancellation - separate data-set (payload) through separate end-point. Consequently, to cancel separate lot, platform will be need to use to operate with a payload based on 'lot cancellation command-model' (same as tender cancellation)

During tendering period

To be provided

During awarding period

To be provided

Tender cancellation

Cancellation of the active stage of contracting process is executed within submitting of separate object, contains all the information needed to execute and publish such cancellation.

Extension of initial 'Process Stage' data-model

If submitted amendment of tender cancellation exists, initial query-model of current stage will be extended with additional object - 'amendments':

```
{
 "tag":                                                    "tenderCancelation",
 "tender":                                                                  {
   "amendments":                                                            [
     {
       "id":                                                                "",
       "date": "",
       "type": "cancellation",
       "relatesTo":                                                "tender",
       "amendsReleaseID":                                                  "",
       "rationale":                                                        "",
       "description":                                                      "",
       "documents": [],
       "status":                                                   "pending"
     }
   ]
 }
}
```

Command Model

To submit an amendment for tender cancellation standard command-model should be used for preparation of POST-request for BPE.

Schema

rationale		string	required
Reason for cancellation (selected from a standardized closed value-list)			
description		string	optional
The additional information about cancellation			
documents	array	object	optional
All documents and attachments related to the decision on cancellation See Document			

Example

```
{
  "amendments": [
    {
      "rationale":                                                          "",
      "description":                                                        "",
      "documents":                                                            [
        {                    "$ref":                  "#/definitions/Document"                  }
      }
```

```
      ]
    ]
}
```

Once review period for this decision expired and no claim satisfied, CA is able to switch previously published amendment to the final status and cancel related lots. This will add an additional set of attributes (or amend existing) and switch related lots to the ‘cancelled’ statuses:

```
{
 "tag":                                                    "tenderAmendment",
 "tender":                                                                  {
   "amendments":                                                            [
     {
       "id":                                                                "",
       "date":                                                              "",
       "amendsReleaseID":                                                   "",
       "status": "active",
       "statusDetails":                                          "cancellation"
     }
   ]
 }
}
```

Future steps

In a next release it is considered to use the same approach as for tender cancellation - separate data-set (payload) through separate end-point. Consequently, to cancel separate lot, platform will be need to use to operate with a payload based on ‘lot cancellation command-model’ (same as tender cancellation)

4.2.2 Data models mind-map

Using data-models described above, procurement process itself as well as ‘collaboration’ of its different part could be shown as a structure of building blocks:

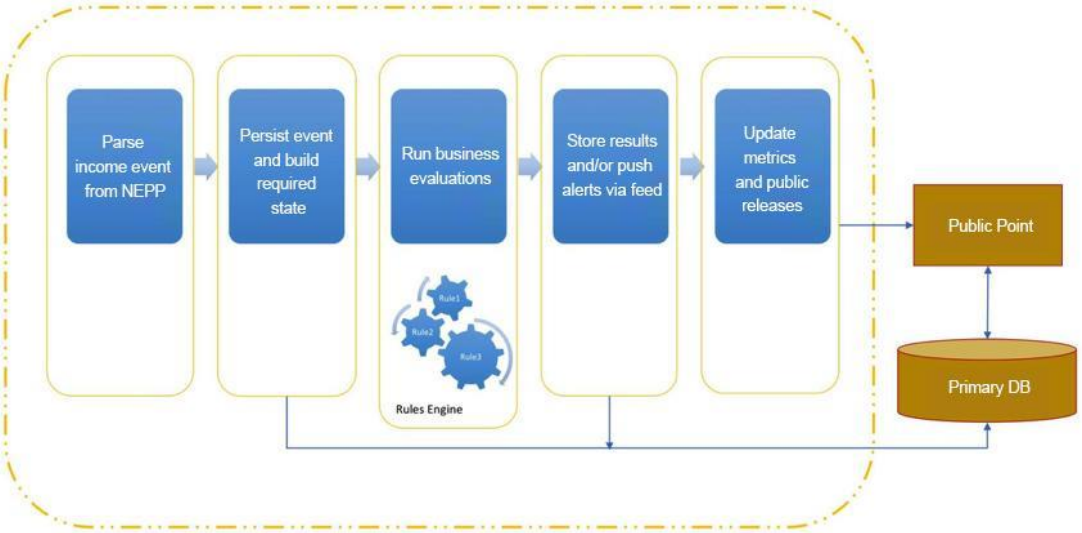


Figure 14. General scheme of the contracting process described using data-models

5 Components' management

These Components shall automate entire procurement lifecycle (from planning to payment) for different procurement methods described below (see "Coverage" chapter). The following figure contains a generic distinction for management of the public procurement process and the interaction between the CAs, EOs and the system itself:

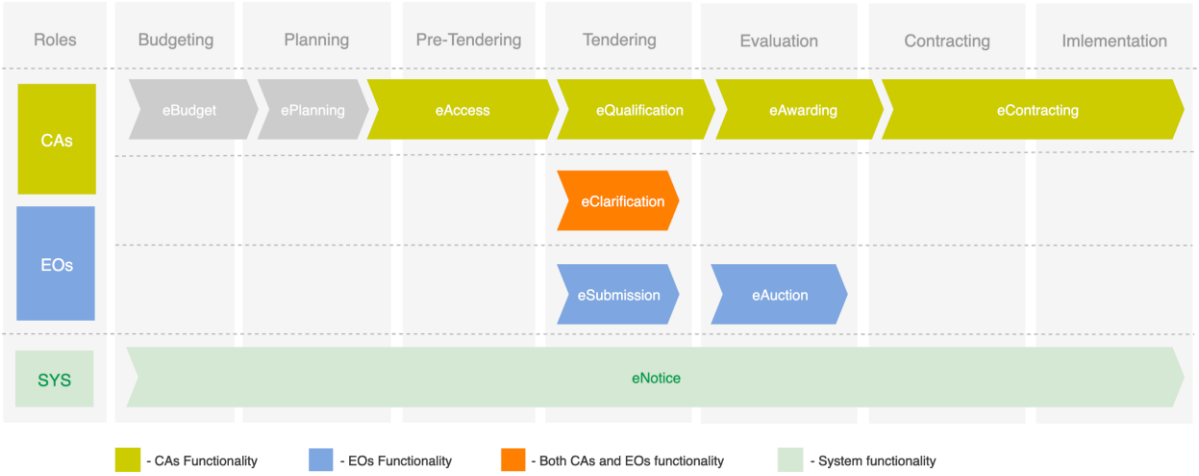


Figure 16. BPE Component diagram as a generic distinction for management of the public procurement process

5.1 eBudget

eBudget allows the online definition and preparation of expenditure items, funding sources, periods of budgeting and budgets in structured form.

5.1.1 Methods

eBudget component provides the following methods which allow to do some actions:

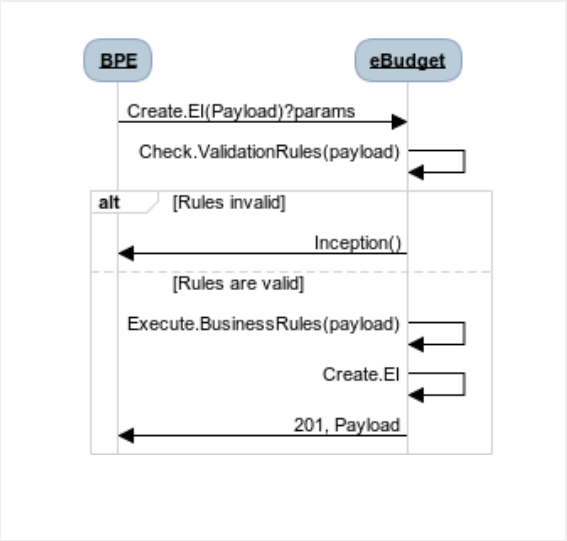
Name	Method	Description	Note
`Create.EI()`	POST	create of new 'Expenditure Item'	
`Update.EI()`	POST	update of existing 'Expenditure Item'	
`Cancel.EI()`	POST	update of existing 'Expenditure Item'	TBD
`Create.FS()`	POST	create of new 'Funding source'	

`Update.FS()`	POST	update of existing ‘Funding source’	
`Check.FS()`	POST	check on existing ‘Funding source’	
`Check.BB()`	POST	check budget breakdown for specific OCID	TBD
`Cancel.FS()`	POST	update of existing ‘Funding source’	TBD
`Get.Buyer()`	GET	get information about ‘buyer’ for specific OCID	TBD

Create.EI()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules,
- creation a 'EI' entity and
- generation of a number of related attributes identifiers
- subsequent ownership and management of the created entity, its life cycle and possible changes




```
POST /ei?ownerID=...&country=...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
ownerID	string	NEPP ID	mandatory
country	string	Code of country according to countries codelist	mandatory
date	date-time	Registered date of request	mandatory
data	object	bpe-payloads/eBudget/createEI-request.json	mandatory

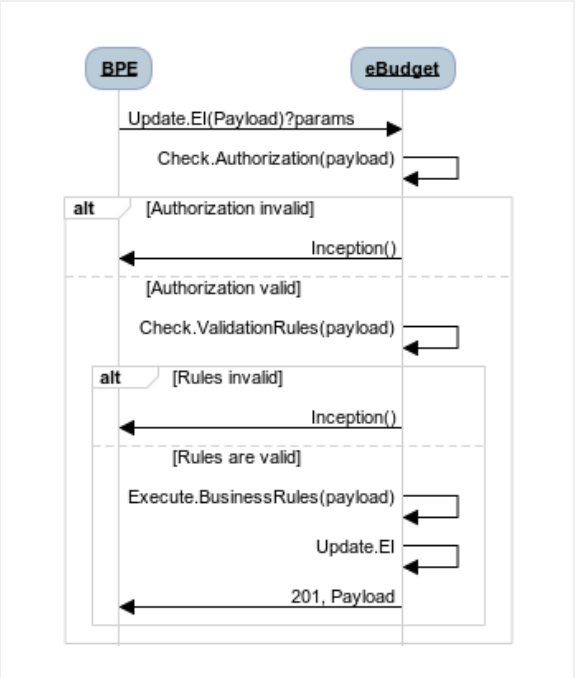
Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eBudget/createEI-resopnse.json	mandatory

Update.EI()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules,
- update of existing 'EI' entity and
- generation of a number of related attributes




```
POST /ei?ownerID=...&token=...&cpid=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
ownerID	string	NEPP ID	mandatory
token	string	EI's token	mandatory
cpid	string	EI unique identifier	mandatory
data	object	bpe-payloads/eBudget/updateEI-request.json	mandatory

Outcomes

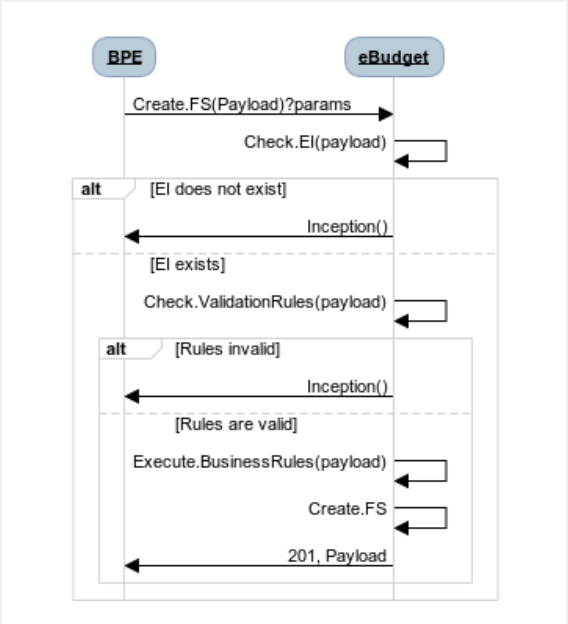
Name	Type	Description	Obligation

success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eBudget/createEI-response.json	mandatory

Create.FS()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules,
- creation of the 'FS' entity
- generation of a number of related attributes and identifiers
- subsequent ownership and management of the created entity, its life cycle and possible changes




```
POST /fs?cpid=...?ownerID=...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
ownerID	string	NEPP ID	mandatory
cpid	string	Parent EI unique identifier	mandatory
date	date-time	Registered date of request	mandatory
data	object	bpe-payloads/eBudget/createFS-request.json	mandatory

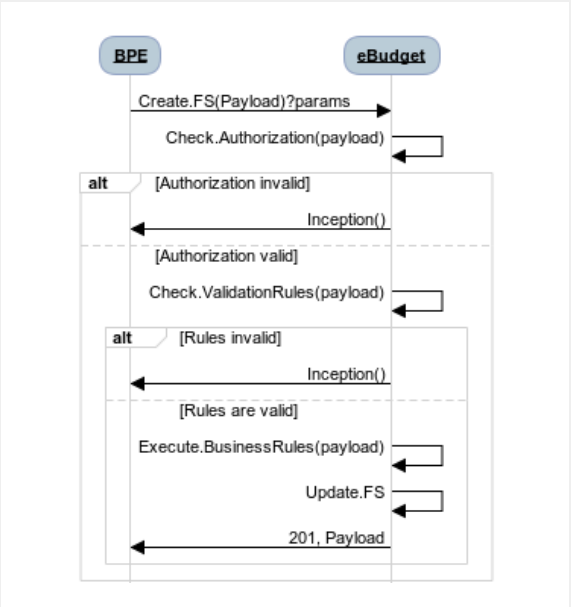
Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eBudget/createFS-response.json	mandatory

Update.FS()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules,
- update of the 'FS' entity
- generation of a number of related attributes




```
POST /fs?ownerID=...&token=...&cpid=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
ownerID	string	NEPP ID	mandatory
token	string	EI's token	mandatory
cpid	string	EI unique identifier	mandatory

data	object	bpe-payloads/eBudget/updateFS-request.json	mandatory

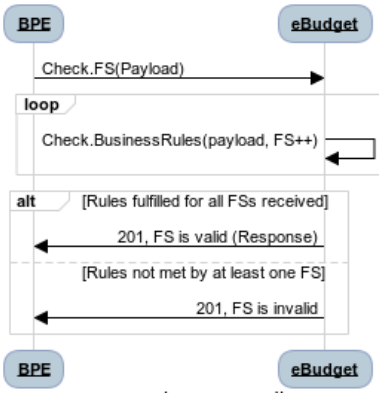
Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eBudget/createFS-response.json	mandatory

Check.FS()

This method is responsible for:

- check of availability of FS with particular ID or array of such FSs for particular procurement category
- extraction of the initial amounts of funds available on specified FSs for specified need




```
POST /fs/check

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
data	object	bpe-payloads/eBudget/checkFS-request.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory

`data`	object	bpe-payloads/eBudget/checkFS-response.json	mandatory

5.1.2 Related entities

All entities that are managed by this module are described below.

Title	Description
Tender	EI setting
Budget	Related budget
Address	EI buyer address and details
Classification	Budget classification
ContacPoint	EI buyer contact point
Details	Buyer details
EuropeanUnionFunding	Project description
Identifier	Auxiliar entity to declare identifiers
Period	Budget or tener period devinition
Person	Person declaration form buyers
Value	Auxiliar entity for vaules
_Enums	Domain values.
EIEntity	
FSEntity	

5.1.3 Functional relationship

This section does not have functional relationship.

5.2 ePlanning

ePlanning component allows CAs to construct their procurement plans by scheduling of procedures during a defined period. The ePlanning functionality grants the possibility of aggregating all Annual Procurement Plans by different criteria such as CPV codes, procurement procedure types, dates, and others to be defined in order to crosscheck the planning between contracting authorities and build an annual aggregated procurement plan, and facilitates the aggregation of procurements. Also ePlanning allows the identification of potential individual procurement plans to be aggregated in order to launch a repetitive procedure (i.e. FA) that achieves gains in terms of efficiency and savings.

5.2.1 Methods

ePlanning component includes following methods:

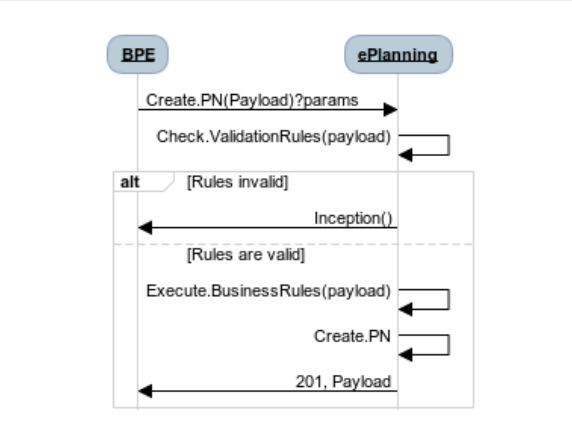
Name	Method	Description
`Create.PN()`	POST	create of new ‘Periodic Notice’

`Update.PN()`	POST	create of new 'Periodic Notice'
`Check.PN()`	GET	check whether specified entity of PN exists

Create.PN()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules,
- creation of the 'PN' entity
- generation of a number of related attributes and identifiers
- subsequent ownership and management of the created entity, its life cycle and possible changes




```
POST /pn?stage=PN&country=...&pmd=...&ownerID=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
`ownerID`	string	ID of Platform	mandatory
`stage`	string	Code of stage	mandatory
`country`	string	ISO of Country according to codelist	mandatory
`pmd`	string	Procurement Method Details according to codelist	mandatory
`date`	date-time	Date of the initiation of the operation	mandatory
`data`	object	bpe-payloads/eAccess/createPN-request.json	mandatory

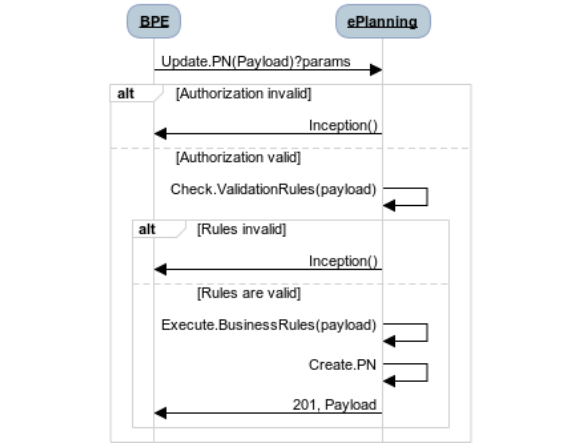
Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAccess/createPN-response.json	mandatory

Update.PN()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules,
- update of the 'PN' entity
- generation of a number of related attributes




```
POST                                                    /pn?ownerID=...&token=...&date=...

{} // payload according to internal command model

201                                                                                  OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
ownerID	string	ID of Platform	mandatory
date	date-time	Date of the initiation of the operation	mandatory
token	string	PNs' token	mandatory

`data`	object	bpe-payloads/eAccess/updatePN-request.json	mandatory

Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory
`data`	object	bpe-payloads/eAccess/updatePN-response.json	Mandatory

5.2.2 Related entities

Pending content.

5.2.3 Functional relationship

This component is described in the functional document "EBRD_MTender_Technical Specifications_v16.0.docx", in section 4.2.

5.3 eAccess

This component is responsible for receiving, validation and saving of core information about the procurement according to selected procurement method, geography, etc (i.e. title, value, CPV code, terms of reference, duration, qualification criteria, award criteria, etc.) and it amendments. Apart from the initiation of a new Contracting Process, the eAccess includes the links to all related processes (such as definition of used budget or conducted prior notices).

5.3.1 Methods

eAccess component includes following methods:

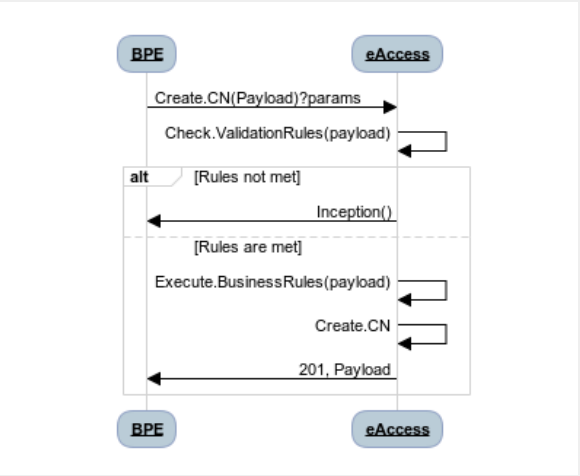
Name	Method	Description
`Create.CN()`	POST	Create new 'Contract Notice'
`Update.CN()`	POST	Update existing 'Contract Notice'
`Update.Tender()`	POST	Suspension/resumption of the flow
`Terminate.Tender()`	POST	Termination of all contracting process

`Terminate.Lot()`	POST	Termination of separate lot
`Get.Lots()`	GET	Receive a list of lots in ‘active’ status

Create.CN()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules
- creation of the ‘CN’ entity
- generation of a number of related attributes and identifiers
- subsequent ownership and management of the created entity, its life cycle and possible changes




```
POST /cnonpn?cpid=...&ownerID=...&token=...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
`ownerID`	string	ID of Platform	mandatory
`token`	string	PNs’ token	mandatory
`cpid`	string	Contracting process ID	mandatory
`date`	date-time	Date of the initiation of the operation	mandatory
`data`	object	bpe-payloads/eAccess/createCNonPN-request.json	mandatory

Outcomes

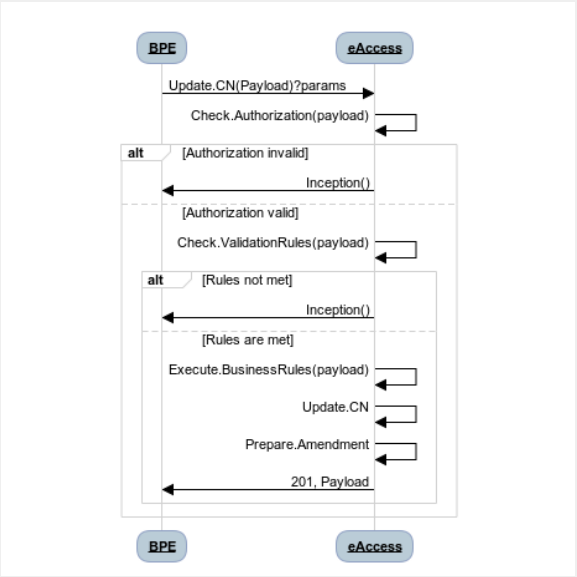
Name	Type	Description	Obligation

success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAccess/createCNonPN-response.json	mandatory

Update.CN()

This method is responsible for:

- validation the obtained data (payload) according to the prescribed rules,
- update of the 'CN' entity
- generation of a number of related attributes
- generation of separate item for the 'amendments' array for each successful update of each particular CN




```
POST /cn?cpid=...&ownerID=...&token=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
ownerID	string	ID of Platform	mandatory
token	string	CN's token	mandatory
cpid	string	Contracting process ID	mandatory
data	object	bpe-payloads/eAccess/updateCN-request.json	mandatory

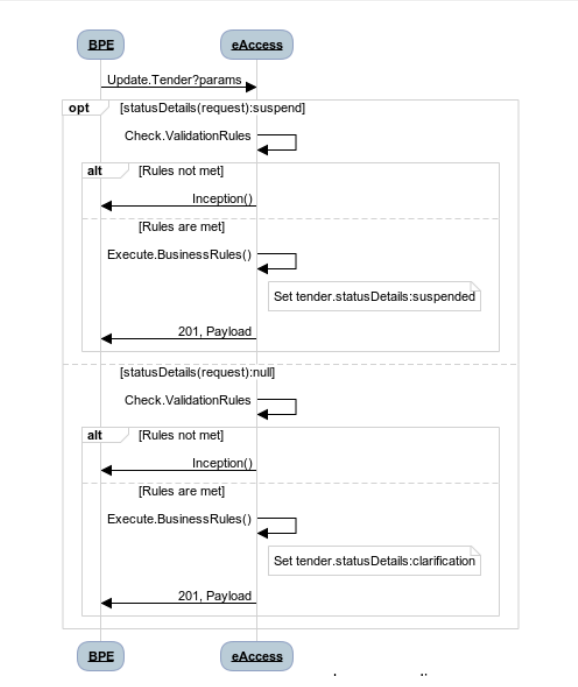
Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAccess/updateCN-response.json	mandatory

Update.Tender()

This method is responsible for:

- Suspension of the contracting process according to the general flow prescriptions
- Resumption of the contracting process according to the general flow prescriptions




```
POST /cn?cpid=...&statusDetails=...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
date	string	Date of the initiation of the operation	mandatory

statusDetails	string	SUSPEND || NULL	mandatory

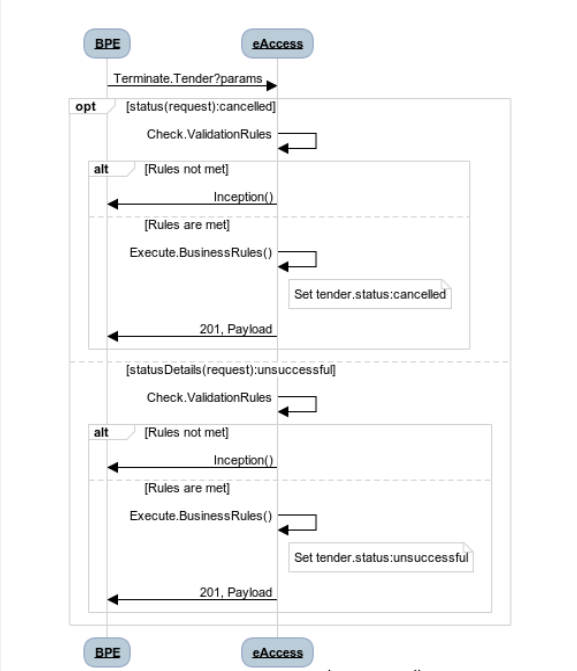
Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAccess/updateTender-response.json	mandatory

Terminate.Tender()

This method is responsible for management of the *status* attribute of the Contracting Process in case of negative flow

- Set *tender.status:unsuccessful* for for those contracting processes where minimum number of offers not achieved
- Set *tender.status.cancelled* for those contracting processes where such process cancelled by CA himself




```
POST /cn?cpid=...&status=...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory

`date`	string	Date of the initiation of the operation	mandatory
`status`	string	CANCELED | UNSUCCESSFUL	mandatory

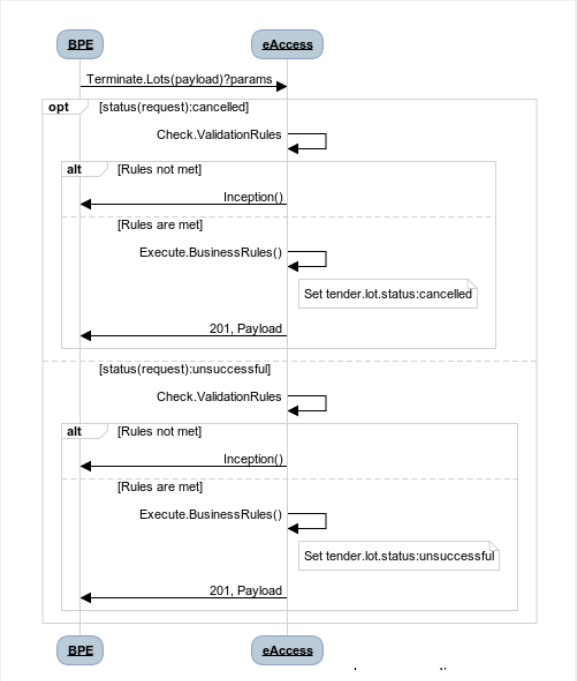
Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory
`Data`	object	bpe-payloads/eAccess/terminateTender-response.json	mandatory

Terminate.Lots()

This method is responsible for management of the *status* attribute of the separate lots of the Contracting Process in case of negative flow

- Set *tender.lot.status:unsuccessful* for for those lots under contracting process where minimum number of offers not achieved
- Set *tender.lot.status.cancelled* for those lots under contracting process where such lot cancelled by CA himself




```
POST /lots?cpid=...&status=...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation

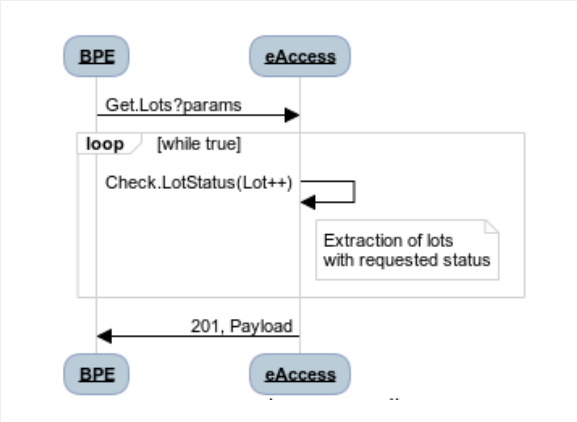
cpid	string	Contracting process ID	mandatory
date	string	Date of the initiation of the operation	mandatory
status	string	UNSUCCESSFUL | CANCELED	mandatory
data	object	bpe-payloads/eAccess/terminateLots-request.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAccess/terminateLots-response.json	mandatory

Get.Lots()

This method is responsible for extraction of the set of lots in needed status




```
GET /lots?cpid=...&status=...

{} // payload according to internal command model

200                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
status	string	ACTIVE || UNSUCCESSFUL || CANCELLED	mandatory

Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory
`data`	object	bpe-payloads/eAccess/getLots-response.json	mandatory

5.3.2 Related entities

All entities that are managed by this module are described below:

Title	Description
AcceleratedProcedure	
Address	Procuring Entity address description
Budget	Related budget
BudgetBreakdown	
Classification	Budget classification
ContacPoint	Procuring Entity contact point
ContractPeriod	Tender initial and ends
DesignContest	
Document	Tender documents declaration
Details	Buyer details
DynamicPurchasingSystem	
ElectronicAuctions	It includes ElectronicAuctionsDetails and ElectronicAuctionModalities
ElectronicWorkflows	To declare the type of workflows thar a process has to follow.
EuropeanUnionFunding	Project description
Framework	To declare if a process is a Framework
Identifier	Auxiliar entity to declare identifiers
Item	A product or service
JointProcurement	Type of procurement
Lot	Each or a tender lots
LotGroup	Group for lots
Period	Budget or tener period devinition
PlaceOfPerformance	
Planning	Organization plan with the budget
ProcedureOutsourcing	
RecurrentProcurement	Flag for recurrent processes
Renewal	
SourceParty	
Tender	
TenderProcess	It includes the Amendment
Unit	Auxiliar entity to describe the items
Value	Auxiliar entity for economic values
Person	Person declaration form buyers
Value	Auxiliar entity for vaules
Variant	

5.3.3 Functional relationship

This component is described in the functional document “EBRD_MTender_Technical Specifications_v16.0.docx”, in section 4.3.

5.4 eClarification

This component provides the option to ask questions and receive answers regarding the conditions of particular procurement as well as allow CAs to answering questions from EOs.

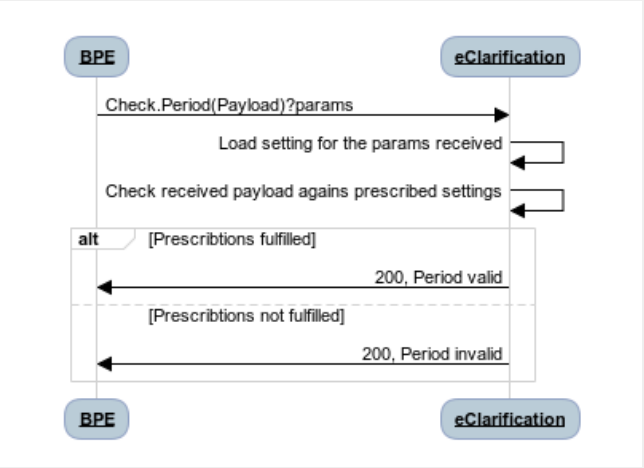
5.4.1 Methods

eClarification component includes following methods:

Name	Method	Description
`Check.Period()`	GET	Validate a validity of the enquiry period for specific landscape
`Save.Period()`	POST	Establish clarification period under Contracting Process
`Create.Enquiry()`	POST	Create new enquiry
`Update.Enquiry()`	POST	Update existing enquiry with an answer
`Check.Enquiries()`	GET	Check whether all enquiries complete with an answers

Check.Period()

This method is responsible for the validation of requested period for clarifications against prescribed setting for this 'landscape'




```
GET /period?&stage=...&startDate...&endDate=...&pmd=...&country=...

{} // payload according to internal command model

200                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
`startDate`	date-time	tenderPeriod.startDate	mandatory
`endDate`	date-time	tenderPeriod.startDate	mandatory
`pmd`	string	Code of applied procurement method	mandatory
`country`	string	Code of country of initiation	mandatory

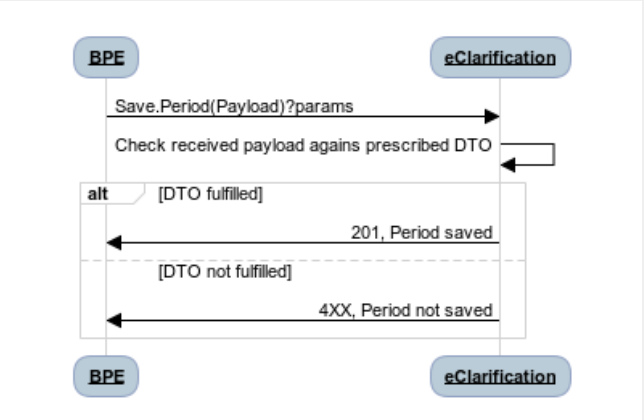
Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory

Save.Period()

This method is responsible for:

- validation of requested period for clarifications against prescribed data format
- establishing clarification period for the retaled contracting process




```
POST /period?cpid=...&stage=...&startDate...&endDate=...&pmd=...&country=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
`cpid`	string	Contracting process ID	mandatory
`startDate`	date-time	tenderPeriod.startDate	mandatory
`endDate`	date-time	tenderPeriod.startDate	mandatory
`pmd`	string	Code of applied procurement method	mandatory
`country`	string	Code of country of initiation	mandatory

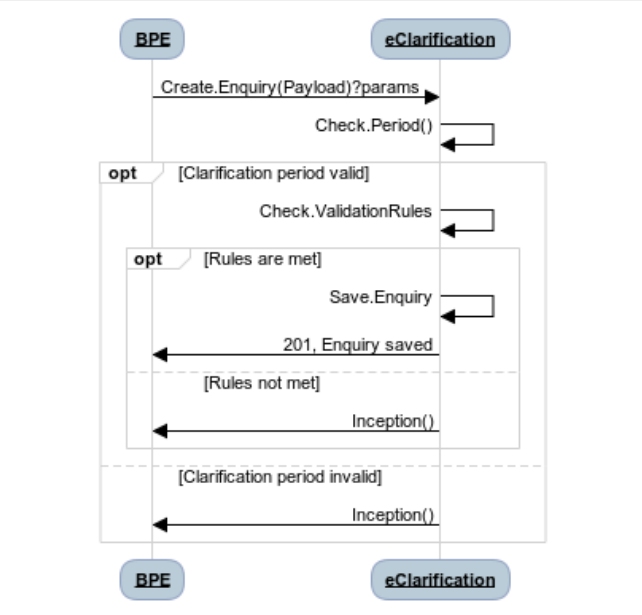
Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory
`data`	object	bpe-payloads/eClarification/savePeriodResponse.json	mandatory

Create.Enquiry()

This method is responsible for:

- Validation of general ability of the clarifications under required contracting process
- Validation of the business rules prescribed for the clarification request under requested contracting process
- Save valid request for clarification




```
POST /enquiry?cpid=...&stage=...&OwnerID...&date=...

{} // payload according to internal command model
```

```
201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
date	date-time	Date of initiation of this enquiry	mandatory
ownerID	string	ID of Platform	mandatory
data	object	bpe-payloads/eClarification/createEnquiryPayload.json	mandatory

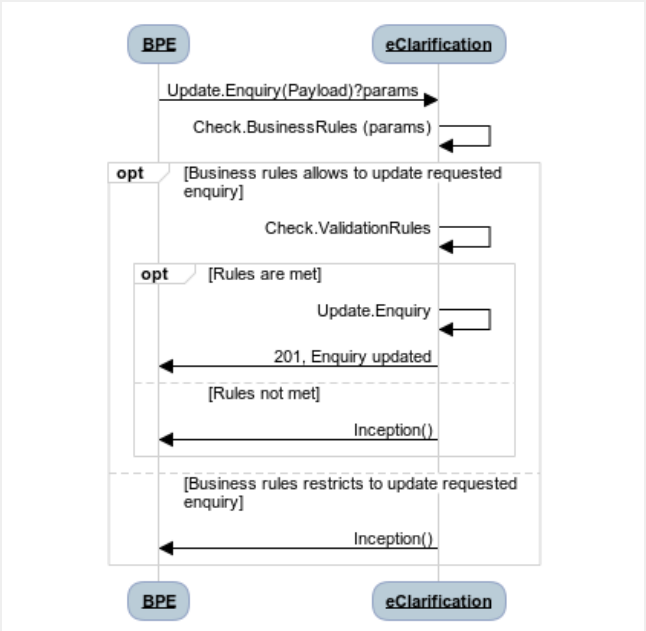
Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eClarification/createEnquiryResponse.json	mandatory

Update.Enquiry()

This method is responsible for:

- Verification request against prescribed business rules (if requested enquiry is available for update considering current state of the contracting process)
- Validation of the rules prescribed for the clarification under requested contracting process
- Save valid clarification



```
POST /enquiry?cpid=...&OwnerID...&date=...&token=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

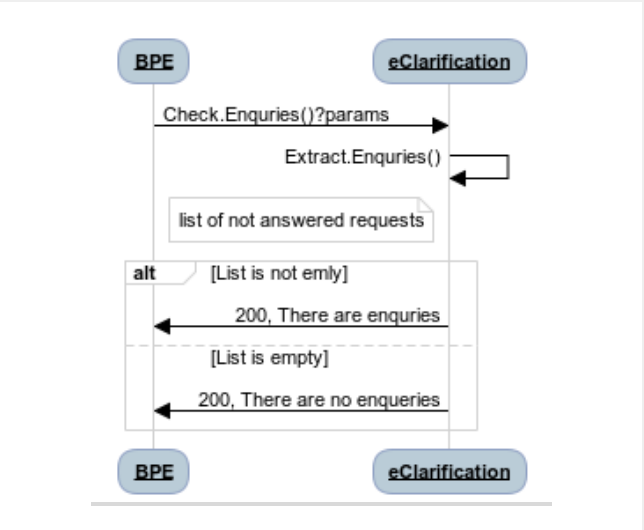
Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
date	date-time	Date of publication of this answer	mandatory
ownerID	string	ID of Platform	mandatory
token	string	Token for this enquiry	mandatory
data	object	bpe-payloads/eClarification/updateEnquiryPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eClarification/updateEnquiryResponse.json	mandatory

Check.Enquiries()

This method is responsible for check if there is at least one clarification request with no related respond



```
GET /enquiry?cpid=...&stage=...&OwnerID...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
ownerID	string	ID of Platform	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory

5.4.2 Related entities

All entities that are managed by this module are described below:

Title	Description
Address	Needed to describe the enquiry author details
Tender	It allows to check the enquiry period
Enquiry	A new entity is registered in the system for its lot.
ContacPoint	Needed to describe the enquiry author contact point
Details	Needed to describe the author details
Identifier	Needed to describe the author identifies
OrganizationReference	To describe enquiry author
Period	Needed to describe the Tender enquiry period

5.4.3 Functional relationship

This component is described in the functional document “EBRD_MTender_Technical Specifications_v16.0.docx”, in section 4.5.

5.5 eSubmission

This component allows EOs to respond to Contract Notice by preparing their offers in a structured and secure way, and submit their offers or electronically to a CA, including using templates created by the CA.

On the CAs side, this component generates a standardized schema for bid or proposal in the relevant procurement method, stores received bids or proposal until expiry of tender deadlines and allows secure opening of the received tenders upon expiry of the tender deadlines.

5.5.1 Methods

eSubmission component includes following methods:

Name	Method	Description
`Check.Period()`	GET	Check validity of submission period for requested conditions
`Save.Period()`	POST	Establish submission period for contracting process
`Create.Bid()`	POST	Create new bid
`Update.Bid()`	POST	Update existing bid
`Update.BidStatusDetails()`	POST	Update *statusDetails* value according to the process flow
`Set.BidStatusDetails()`	POST	Set *statusDetails* value according to the process flow
`Get.Bids()`	GET	Get list of bids registered under specific contracting process
`Finalize.ValidBids()`	POST	Move all valid bids to the final state according to the flow

Check.Period()

This method is responsible for the validation of requested period for submission against prescribed setting for this contracting process considering geography and applied legal basic


```
GET /period?country=...&pmd=...&stage=...&startDate...&endDate=...
```

```
{} // payload according to internal command model

200                                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
`country`	string		mandatory
`pmd`	string		mandatory
`startDate`	date-time		mandatory
`endDate`	date-time		mandatory

Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory

Save.Period()

This method is responsible for:

- validation of requested period for submission against prescribed data format
- establishing submission period for the retaled contracting process

Sequence diagram

```
POST /period?cpid=...&stage=...&startDate...&endDate=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
`cpid`	string	Contracting process ID	mandatory
`startData`	date-time		mandatory
`endDate`	date-time		mandatory

Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory
`data`	object	bpe-payloads/eSubmission/savePeriodResponse.json	mandatory

Create.Bid()

This method is responsible for:

- Validation of general ability of the submission under required contracting process
- Validation of the business rules prescribed for the bid request under requested contracting process
- Save valid bid

Sequence diagram

```
POST /bid?cpid=...&stage=...&OwnerID=...&date=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
ownerID	date-time		mandatory
date	date-time		mandatory
data	object	bpe-payloads/eSubmission/createBidPayload.json	

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eSubmission/createBidResponse.json	mandatory

Update.Bid()

This method is responsible for:

- Verification request against prescribed business rules (if requested bid is available for update considering current state of the contracting process)
- Validation of the rules prescribed for the submission under requested contracting process
- Save valid updated bid

Sequence diagram

```
POST /bid?cpid=...&stage=...&OwnerID=...&date=...&token=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
ownerID	date-time		mandatory
date	date-time		mandatory
token	string		mandatory
data	object	bpe-payloads/eSubmission/updateBidPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eSubmission/updateBidResponse.json	mandatory

Update.BidStatusDetails()

This method is responsible for switching bids collected under specific contracting process where such change requested by BPE according to the current state of the contracting process to the relevant temporary status


```
POST /bid?cpid=...&stage=...&ownerID=...&date=...&token=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
date	date-time		mandatory
data	object	bpe-payloads/eSubmission/updateBidStatusDetailsPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eSubmission/updateBidStatusDetailsResponse.json	mandatory

Set.BidStatusDetails()

This method is responsible for switching bids collected under specific contracting process where award decision took place against such bids to the relevant temporary status


```
POST /bid?cpid=...&awardStatus=...&token=...

{} // payload according to internal command model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
bidId	date-time		mandatory
awardStatus	string	UNSUCCESSFUL | ACTIVE	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
data	object	bpe-payloads/eSubmission/setBidStatusDetailsResponse.json	mandatory

Get.Bids()

This method is responsible for extraction a set of bids collected under specific contracting process and available for disclosure where award period has been started


```
GET /bid?cpid=...&pmd=...&country=...&status=...

{} // payload according to internal command model

200                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
pmd	date-time	Code of applied procurement method	mandatory

`country`	string	Code of the country of initiation of contracting process	mandatory
`status`	string	pending	mandatory

Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory
`data`	object	bpe-payloads/eSubmission/getBidsResponse.json	mandatory

Finalize.ValidBids()

This method is responsible for switching bids collected under specific contracting process where award decision took place against such bids to the relevant permanent status


```
POST /bid?cpid=...&stage=...&date=...

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
`cpid`	string	Contracting process ID	mandatory
`date`	string	Date of initiation of finalization	mandatory

Outcomes

Name	Type	Description	Obligation
`success`	boolean	TRUE || FALSE	mandatory

`data`	object	bpe-payloads/eSubmission/finalizeValidBidsResponse.json	mandatory

5.5.2 Related entities

All entities that are managed by this module are described below:

Title	Description
AccountIdentification	
AdditionalAccountIdentifier	
Address	Is completed with AddressDetails, data class CountryDetails, RegionDetails, LocalityDetails
BankAccount	
Bid	A tender bid
Bids	List of Bid
BusinessFunction	It is completed with Period and Document
ContactPoint	It describes the tenderers contact details
Details	It completes the tenderer description
Document	It completes the tender documents
Identifier	It is used for tenderer identifies
IssuedBy	
IssuedThought	
LegalForm	
OrganizationReference	Describes the tenderers
Period	The tender star and end date
Permit	
PermitDetails	
Persone	
Requirement	
RequirementResponse	
ValidityPeriod	The tender start and end validity period
Value	Auxiliar entity to describe economic values
_Enums.kt	List of values

5.5.3 Functional relationship

This component is described in the functional document “EBRD_MTender_Technical Specifications_v16.0.docx”, in section 4.6.

5.6 eQualification

This component handles legal, economic and quality qualification of the tenderers and its proposals. It is responsible for accessing EOs master data. It is also used for shortlist management of procedures that allow shortlisting prior to submission of tenders.

5.6.1 Methods

eQualification component includes following methods:

Name	Method	Description

Create.Awards()	POST
Update.Award()	POST
End.AwardPeriod()	POST
Finalize.Awards()	POST

Create.Awards()

This method is responsible for:

- Analysis of the payload received in order to establish awarding processes for those lots where enough bids were collected and current status of such lots allows to start evaluation
- Generation of the 'awards' related to the lots under evaluation and related bids
- Initial ranking of the generated awards according to applied awarding strategy

Sequence diagram

```
POST /qualification?stage=...&country=...&pmd=...&OwnerID=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
stage	string	Code of stage	mandatory
cpid	string		
country	string	ISO of Country according to codelist	mandatory
pmd	string	Procurement Method Details according to codelist	mandatory
startDate	date-time		mandatory

payload	object	bpe-payloads/eAccess/createAwardsPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
responseDetails	array	Description for the code of response (if applicable)	mandatory
data	object	bpe-payloads/eAccess/createAwardsResponse.json	mandatory

Update.Awards()

This method is responsible for switching ‘awards’ established for the evaluation needs under specific contracting process where award decision was made to the relevant temporary status


```
POST /qualification?cpid=...&stage=...&date=...&OwnerID=...&token=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
stage	string	Code of stage	mandatory
cpid	string		
owner	string	ISO of Country according to codelist	mandatory
token	string	Procurement Method Details according to codelist	mandatory
date	date-time		mandatory

payload	object	bpe-payloads/eAccess/updateAwardsPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
responseDetails	array	Description for the code of response (if applicable)	mandatory
data	object	bpe-payloads/eAccess/createAwardsResponse.json	mandatory

End.AwardPeriod()

This method is responsible for completion of the awarding period for specific lot where award decision was made and stand-still period expired


```
POST /qualification?cpid=...&stage=...&OwnerID=...&token=...&endDate=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
stage	string	Code of stage	mandatory
cpid	string		
owner	string	ISO of Country according to codelist	mandatory
token	string	Procurement Method Details according to codelist	mandatory
endDate	date-time	дата завершения периода	mandatory

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
responseDetails	array	Description for the code of response (if applicable)	mandatory
data	object	bpe-payloads/eAccess/endAwardPeriodResponse.json	mandatory

Finalize.Awards()

This method is responsible for switching ‘awards’ where awarding period has been closed to the relevant permanent status


```
POST /qualification?cpid=...&stage=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
stage	string	Code of stage	mandatory
cpid	string		

Outcomes

Name	Type	Description	Obligation
success	boolean	TRUE || FALSE	mandatory
responseDetails	array	Description for the code of response (if applicable)	mandatory

data	object	bpe-payloads/eAccess/finalizeAwardsResponse.json	mandatory

5.6.2 Related entities

All entities that are managed by this module are described below:

Title	Description
Address	Incules AddressDetails, CountryDetails, RegionDetails and LocalityDetails
Award	Tender award
Bid	Tender bid
Classification	Tender classification cirteria
ContactPoint	Award suppliers details
Details	
Document	Tender documents
Identifier	Auxiliar entitiy to intentify a bider
Item	A product or service
Lot	
OrganizationReference	Tender procuring entity
Period	Tender start and end date
Unit	Item unit
Value	Auxiliar entity to economic values
_Enums.kt	List of values

5.6.3 Functional relationship

This component is described in the functional document “EBRD_MTender_Technical Specifications_v16.0.docx”, in section 4.7.

5.7 eAuction

This component facilitates the configuration and management of reverse auction held electronically.

5.7.1 Methods

eAuction component includes following methods:

Name	Method	Description
Validate.Auction()	GET	
Schedule.Auction()	POST	

Cancel.Auction()	POST
Run.Auction	POST

Validate.Auction()

This method is responsible for validation of the expected start date of eAuction requested within preparation of the Contract Notice of the specific contracting process


```
GET /auction/check

{} // payload according to internal request model

200                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
data	object	bpe-payloads/eAuction/validateAuctionPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory

Schedule.Auction()

This method is responsible for scheduling of eAuction for each lot under specific contracting process


```
POST /auction?cpid=...&ocid=...

{} // payload according to internal request model
```

```
201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
data	object	bpe-payloads/eAuction/scheduleAuctionPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory
data	string	bpe-payloads/eAccess/scheduleAuctionResponse.json	mandatory

Cancel.Auction()

This method is responsible for cancellation of the previously scheduled eAuction under specific contracting process

Sequence diagram

```
POST /auction?cpid=...&ocid=...&lotID=...status=cancelled&token=...

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory

ocid	string		mandatory
lotID	string		mandatory
status	string	CANCELLED	mandatory
token	string		mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory

Run.Auction()

This method is responsible for launch of eAuction for the separate lot under specific contracting process


```
POST /auction?cpid=...&ocid=...&lotID=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
data	object	bpe-payloads/eAuction/runAuctionPayload.json	mandatory

Outcomes

success	string		mandatory
data	string	bpe-payloads/eAccess/runAuctionResponse.json	mandatory

When an electronic auction ends due to success execution, eAuction component automatically sends relevant notification to BPE. Such response contents all statistics and results of electronic auctions under all lots in this specific contracting process.

Result

Name	Type	Description	Obligation
success	string		mandatory
cpid			mandatory
ocid			mandatory
auctionResults	string	bpe-payloads/eAccess/auctionResult.json	mandatory

5.7.2 Related entities

All entities that are managed by this module are described below:

Title	Description
Amount	
Auction	The auction process
Bid	A presented bid
Breakdown	
Bucket	
Command	
Country	
cpid	
Currency	
Date	
lotId	
Modality	
Offer	
operationId	
Period	
PlatformId	
progressId	
result	
sign	
slots	
tender	
value	
version	

5.7.3 Functional relationship

This component is described in the functional document "EBRD_MTender_Technical Specifications_v16.0.docx", in section 4.8.

5.8 eAwarding

This component allows preparation of the Contract Award Notice and Awarded Contract itself in a structured way. It also ensures exchange of documents with the tenderer during the awarding phase. eAwarding is responsible for all actions after evaluation is completed until the contract is executed.

5.8.1 Methods

eAwarding component includes following methods:

Name	Method	Description
Create.SuccessfulCAN()	POST	
Create.UnsuccessfullCAN()	POST	
Create.CancelledCAN()	POST	
Update.CAN()	POST	
Confirm.CAN()	POST	
Cancel.CAN()	POST	

Create.SuccessfulCAN()

This method is responsible for generation of the 'Contract Award Notice' related to the specified lot where winner was selected

Sequence diagram

```
POST /can?cpid=...&ocid=...&lotID=...&date=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
date	date-lite		mandatory
lotID	string		mandatory
data	object	bpe-payloads/eAwarding/successfulCANPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory
data	string	bpe-payloads/eAwarding/successfulCANResponse.json	mandatory

Create.UnsuccessfullCAN()

This method is responsible for generation of the ‘Contract Award Notice’ related to the specified lot where winner was not selected


```
POST /can?cpid=...&ocid=...&date=...
```

```
{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
date	date-lite		mandatory
data	object	bpe-payloads/eAwarding/unsuccessfulCANPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory
data	string	bpe-payloads/eAwarding/unsuccessfulCANResponse.json	mandatory

Create.CancelledCAN()

This method is responsible for generation of the 'Contract Award Notice' related to the specified lot where procurement was terminated


```
POST /can?cpid=...&ocid=...&date=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
date	date-lite		mandatory
data	object	bpe-payloads/eAwarding/cancelledCANPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory
data	string	bpe-payloads/eAwarding/cancelledCANResponse.json	mandatory

Update.CAN()

This method is responsible for the update of the 'Contract Award Notice' generated with additional data or attributes' values according to the actions taken by parties under post-award stage of the contracting process


```
POST /can?cpid=...&id=...&token=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory

id	string	Identifier of the specific CAN to be updated	mandatory
token	string		mandatory
data	object	bpe-payloads/eAwarding/updateCANPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory
data	string	bpe-payloads/eAwarding/updateCANResponse.json	mandatory

Confirm.CAN()

This method is responsible for switching of the 'Contract Award Notice' related to the specified lot where winner was not selected to the final permanent status


```
POST /can?cpid=...&ocid=...&id=...&token=...

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
token	string		mandatory
id	string	Identifier of the specific CAN to be updated	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory
data	string	bpe-payloads/eAwarding/confirmCANResponse.json	mandatory

Cancel.CAN()

This method is responsible for cancellation of the previously generated 'Contract Award Notice' related to the specified lot where winner was or was not selected due to Review Body decision


```
POST /can?cpid=...&ocid=...&id=...&token=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
id	string	Identifier of the specific CAN to be updated	mandatory
token	string		mandatory
data	object	bpe-payloads/eAwarding/cancelCANPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string		mandatory
data	string	bpe-payloads/eAwarding/cancelCANResponse.json	mandatory

5.8.2 Related entities

Pending content.

5.8.3 Functional relationship

This component is described in the functional document "EBRD_MTender_Technical Specifications_v16.0.docx", in section 4.10.

5.9 eContracting

This component allows preparation Awarded Contracts in a structured way.

5.9.1 Methods

eAwarding component includes following methods:

Name	Method	Description
Create.Contract()	POST	
Update.Contract()	POST	
Issue.Contract()	POST	
Verify.Contract()	POST	
Cancel.Contract()	POST	
Activate.Contract()	POST	
Terminate.Contract()	POST	

Create.Contract()

This method is responsible for:

- Aggregation information from Contract Award Notice(s) received in order to generate a *contract* related to awarded lot(s)
- Generation of the initial draft of *contract* for awarded lot(s)

Sequence diagram

```
POST /ac?cpid=...&ocid=...&lang=...
```

```
{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
lang	string		mandatory
data	object	bpe-payloads/eAwarding/createContractPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAwarding/createContractResponse.json	mandatory

Update.Contract()

This method is responsible for update of the initial draft of the contract related to awarded lot(s) with additional required data


```
POST /ac?cpid=...&ocid=...&token=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string		mandatory
data	object	bpe-payloads/eAwarding/updateContractPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAwarding/updateContractResponse.json	mandatory

Issue.Contract()

This method is responsible for issuing of the final version of the contract to be signed once all the prearations made


```
POST /issue?cpid=...&ocid=...&country=...

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string	Identifier of the specific CAN to be updated	mandatory
country	string		

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAwarding/issueContractResponse.json	mandatory

Sign.Contract()

This method is responsible for signing contracts by Parties


```
POST /confirm?cpid=...&ocid=...&requestId=...&ownerId=...&date=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string	Identifier of the specific CAN to be updated	mandatory
requestID	string		mandatory
ownerID	string		mandatory
date	date-time		mandatory
data	string	bpe-payloads/eAwarding/signContractResponse.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAwarding/signContractResponse.json	mandatory

Verify.Contract()

This method is responsible for transfer signed contract to the external Body (Treasury) for verification


```
POST /verify?cpid=...&ocid=...

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string		mandatory
ocid	string	Identifier of the specific AC to be updated	mandatory

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory

As a result of success execution, method will switch AC to *statusDetails:verification*

Result

As a result of AC verification in Treasury, this AC will be switched to one of values for *statusDetails* described in the state-chart diagram

Cancel.Contract()

This method is responsible for cancellation of the previously created contract provided that this contract is not yet activated


```
POST /cancel?cpid=...&ocid=...

{} // payload according to internal response model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
ocid	string	Identifier of the specific AC to be updated	mandatory
data	string	bpe-payloads/eAwarding/cancelContractPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAwarding/cancelContractResponse.json	mandatory

As a result of success execution, method will switch AC to *statusDetails:cancellation* and after expiry of reviewPeriod - to *status:cancelled*

Activate.Contract()

This method is responsible for activation of the contract:

- Signed by Parties
- Verified by Treasury (for state funded contracts)


```
POST /activate?cpid=...&ocid=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
ocid	string	Identifier of the specific AC to be updated	mandatory

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAwarding/signContractResponse.json	mandatory

Terminate.Contract()

This method is responsible for termination of the active contract due to end of performance or preliminary termination


```
POST /terminate?cpid=...&ocid=...&owner=...&date=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8

{} // payload according to internal response model
```

Incomes

Name	Type	Description	Obligation
cpid	string	Contracting process ID	mandatory
ocid	string	Identifier of the specific AC to be updated	mandatory
data	string	bpe-payloads/eAwarding/termanateContractPayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
success	string	TRUE || FALSE	mandatory
data	object	bpe-payloads/eAwarding/termanateContractResponse.json	mandatory

5.9.2 Related entities

All entities that are managed by this module are described below:

Title	Description
Address	It includes AddressDetails, CountryDetails, RegionDetails and LocalityDetails
AgreedMetric	It includes Observation and ObservationUnit
Amendment	It includes DocumentAmendment

Award	Tender award
BudgetSource	
Can	
Classification	Tender classification
ConfirmationRequest	It incvludes RequestGroup, Request, RelatedPerson
ConfirmationResponse	ConfirmationResponseValue, Verification,
ContactPoint	Tender awarded entity contact point.
Contract	
ContractedAward	
Details	It includes DetailsSupplier, DetailsBuyer, Permits, PermitDetails, ValidityPeriod, Issue, BankAccount, AccountIdentifier, LegalForm
DocumentAmedment	A tender amedment document
DocumentAward	Tender document award
DocumentContract	
Identifier	Auxiliar entity to identify
Item	A product or service
Milestone	
OrganizationReference	Details for the awarded entity.
OrganizationReferenceBuyer	
OrganizationReferenceSupplier	
Period	Tender start and end dates.
Person	It includes BusinessFunction and DocumentBF
Planning	It includes Implementation, Transaction, ExecutionPeriod, Budget, BudgetAllocation and PlanningBudgetSource
RelatedProcess	
TreasuryData	
Unit	Units for items
Value	Auxiliar entity to economic values
ValueTax	Auxiliar entity for taxes

5.9.3 Functional relationship

This component is described in the functional document “EBRD_MTender_Technical Specifications_v16.0.docx”, in section 4.11.

5.10 eNotice

The eNotice is in charge of generating Notices using the data available in the eAccess and other components. The notices will be generated in a structured way available to download or send to third parties. The module will also allow the creation of all other needed types of notices for all types of procedures based on the tender specifications introduced in the eProcurement system.

5.10.1 Methods

eNotice component includes following methods:

Name	Method	Description

Release.EI()	POST
Update.EI()	POST
Release.FS()	POST
Update.FS()	POST
Release.PN()	POST
Update.PN()	POST
Release.CN()	POST
Update.CN()	POST
Release.CNPN()	POST
Release.CNPIN()	POST
Update.Period()	POST
Cancel.Lot()	POST
Cancel.Tender()	POST
Release.Enquiry()	POST
Update.Enquiry()	POST
Start.AwardPeriod()	POST
Update.Tender()	POST
Update.Award()	POST
End.AwardPeriod()	POST
End.StandStillPeriod()	POST

Release.EI()

Methods responsible for issuing of the new OCDS-record for EI

```
POST /release?cpid=...&operation=createEI&releasedate=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		EI ID	mandatory
operation		createEI	mandatory
releaseDate	date-time	Date of release	mandatory
data	object	bpe-payloads/eNotice/createEIpayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
ocdsRecord	bpe-payloads/eNotice/createEIresponseRecord.json		
ocdsRelease	bpe-payloads/eNotice/createEIresponseRelease.json		

Update.EI()

Methods responsible for issuing of the new OCDS-release for EI

```
POST /release?cpid=...&operation=updateEI&releasedate=...&token=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid	string	EI ID	mandatory
operation	string	updateEI	mandatory
releaseDate	date-time	Date of release	mandatory
data	object	bpe-payloads/eNotice/updateEIpayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/updateEIresponseRelease.json	

Release.FS()

Methods responsible for issuing of the new OCDS-record for FS

```
POST /release?cpid=...&operation=createFS&releasedate=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		EI ID	mandatory
operation		createFS	mandatory
releaseDate	date-time	Date of release	mandatory
data	object	bpe-payloads/eNotice/createFSpayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
ocdsRecord		bpe-payloads/eNotice/createFSpayloadResponseRecord.json	
ocdsRelease		bpe-payloads/eNotice/createFSpayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/createEIpayloadResponseRelease.json	

Update.FS()

Methods responsible for issuing of the new OCDS-release for FS

```
POST /release?cpid=...&operation=updateFS&releasedate=...

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid	string	EI ID	mandatory
operation	string	updateFS	mandatory
releaseDate	date-time	Date of release	mandatory
data	object	bpe-payloads/eNotice/updateFSpayload.json	mandatory

Outcomes

Name	Type	Description	Obligation
ocdsRecord	bpe-payloads/eNotice/updateFSpayloadResponseRecord.json		

Release.PN()

Methods responsible for issuing of the new OCDS-record for PN

```
POST /release?cpid=...&operation=createPN&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory

operation		createPN	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string		mandatory
data	object	bpe-payloads/eNotice/createPNPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRecord		bpe-payloads/eNotice/createMSpayloadResponseRecord.json	
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	
ocdsRecord		bpe-payloads/eNotice/createStagePayloadResponseRecord.json	
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/udpateEIpayloadResponseRelease.json	
ocdsRelease(s)		bpe-payloads/eNotice/udpateFSpayloadResponseRelease.json	

Update.PN()

Methods responsible for issuing of the new OCDS-release for PN

```
POST /release?cpid=...&operation=updatePN&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		updatePN	mandatory

releaseDate	date-time	Date of release	mandatory
stage	string		mandatory

Outcomes

Name	Type	Description	Obligation
ocdsRecord	bpe-payloads/eNotice/createMSpayloadResponseRecord.json		
ocdsRelease	bpe-payloads/eNotice/createMSpayloadResponseRelease.json		
ocdsRecord	bpe-payloads/eNotice/createStagePayloadResponseRecord.json		
ocdsRelease	bpe-payloads/eNotice/createStagePayloadResponseRelease.json		
ocdsRelease	bpe-payloads/eNotice/udpateEIpayloadResponseRelease.json		
ocdsRelease(s)	bpe-payloads/eNotice/udpateFSpayloadResponseRelease.json		

Release.CN()

Methods responsible for issuing of the new OCDS-record for CN

```
POST /release?cpid=...&operation=createCN&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		createCN	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory

data	object	bpe-payloads/eNotice/createCNpayload.json

Outcomes

Name	Type	Description	Obligation
ocdsRecord		bpe-payloads/eNotice/createMSpayloadResponseRecord.json	
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	
ocdsRecord		bpe-payloads/eNotice/createStagePayloadResponseRecord.json	
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/udpateEIpayloadResponseRelease.json	
ocdsRelease(s)		bpe-payloads/eNotice/udpateFSpayloadResponseRelease.json	

Update.CN()

Methods responsible for issuing of the new OCDS-release for CN

```
POST /release?cpid=...&operation=updateCN&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		updateCN	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/updateCNpayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/udpateEIpayloadResponseRelease.json	
ocdsRelease(s)		bpe-payloads/eNotice/udpateFSpayloadResponseRelease.json	

Release.CNPN()

Methods responsible for issuing of the new OCDS-release for CN based on previous PN

```
POST /release?cpid=...&operation=createCNPN&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		createCNPN	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/updateCNpayload.json	

Outcomes

Name	Type	Description	Obligation

ocdsRelease	bpe-payloads/eNotice/createStagePayloadResponseRecord.json
ocdsRelease	bpe-payloads/eNotice/createStagePayloadResponseRelease.json
ocdsRelease	bpe-payloads/eNotice/createMSpayloadResponseRelease.json

Release.CNPIN()

Methods responsible for issuing of the new OCDS-release for CN based on previous PIN

```
POST /release?cpid=...&operation=createCNPIN&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		createCNPIN	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/updateCNpayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRecord.json	
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	

Update.Period()

Methods responsible for issuing of the new OCDS-release for CN where one of the periods is automatically updated

```
POST /release?cpid=...&operation=updateCN&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		updateCN	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/updatePeriodPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/udpateEIpayloadResponseRelease.json	
ocdsRelease(s)		bpe-payloads/eNotice/udpateFSpayloadResponseRelease.json	

Cancel.Lot()

Methods responsible for issuing of the new OCDS-release for CN where one of the lots is cancelled

```
POST /release?cpid=...&operation=cancelLot&releasedate=...&stage=
```

```
{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		cancelLot	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/cancelLotPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/udpateEIpayloadResponseRelease.json	
ocdsRelease(s)		bpe-payloads/eNotice/udpateFSpayloadResponseRelease.json	

Cancel.Tender()

Methods responsible for issuing of the new OCDS-release for CN where all tender is cancelled

```
POST /release?cpid=...&operation=cancelTender&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		cancelTender	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/cancelTenderPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/udpateEIpayloadResponseRelease.json	
ocdsRelease(s)		bpe-payloads/eNotice/udpateFSpayloadResponseRelease.json	

Release.Enquiry()

Methods responsible for issuing of the new OCDS-release for CN where new clarification request is disclosed

```
POST /release?cpid=...&operation=createEnquiry&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory

operation		createEnquiry	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/createEnquiryPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	

Update.Enquiry()

Methods responsible for issuing of the new OCDS-release for CN where new clarification is received

```
POST /release?cpid=...&operation=updateEnquiry&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		updateEnquiry	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/createEnquiryPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/createStagePayloadResponseRelease.json	

Start.AwardPeriod()

Methods responsible for issuing of the new OCDS-release for CN where awardPeriod stars

```
POST /release?cpid=...&operation=startAwarding&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		startAwarding	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/startAwardPeriodPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/startAwardPeriodRelease.json	

Update.Tender()

Methods responsible for issuing of the new OCDS-release for CN where contracting process is suspended/refused

```
POST /release?cpid=...&operation=...&releasedate=...&stage=

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		suspend | unsuspend	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/startAwardPeriodPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease	bpe-payloads/eNotice/createStagePayloadResponseRelease.json		

Update.Award()

Methods responsible for issuing of the new OCDS-release for CN where award decision made

```
POST /release?cpid=...&operation=...&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		updateAward	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/updateAwardpayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/startAwardPeriodRelease.json	

End.AwardPeriod()

Methods responsible for issuing of the new OCDS-release for CN where awardPeriod complete

```
POST /release?cpid=...&operation=...&releasedate=...&stage=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		standStilStart	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/endAwardPeriodPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease		bpe-payloads/eNotice/startAwardPeriodRelease.json	
ocdsRelease		bpe-payloads/eNotice/createMSpayloadResponseRelease.json	

End.StandStillPeriod()

Methods responsible for issuing of the new OCDS-release for CN where stand-still period under awarding exipred

```
POST /release?cpid=...&operation=...&releasedate=...&stage=
```

```
{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
cpid		Contracting process ID	mandatory
operation		standStillEnd	mandatory
releaseDate	date-time	Date of release	mandatory
stage	string	PS | PQ | EV	mandatory
data	object	bpe-payloads/eNotice/endStandStillPeriodPayload.json	

Outcomes

Name	Type	Description	Obligation
ocdsRelease	bpe-payloads/eNotice/createStagePayloadResponseRelease.json		
ocdsRelease	bpe-payloads/eNotice/createMSpayloadResponseRelease.json		

5.10.2 Related entities

All entities that are managed by this module are described below:

Title	Description
AcceleratedProcedure	
Address	Procuring Entity address description
Amendment	Tender amendment
Award	
Bid	Tender bid
Bids	List of bids
BidsStatistic	
Budget	Related budget
BudgetBreakdown	
BudgetSource	
Change	
Classification	Budget classification
Coefficient	

CoefficientValue	
ConfirmationRequest	
ConfirmationResponse	
ContacPoint	Procuring Entity contact point
Contract	
ContractTerm	
Conversion	
Criteria	Tender initial and ends
Criterion	
DesignContest	
Document	Document to publish.
Details	It includes Permits, PermitDetails, BankAccount, AccountIdentifier and LegalForm.
DynamicPurchasingSystem	
ElectronicAuctions	It includes ElectronicAuctionsDetails and ElectronicAuctionModalities
ElectronicWorkflows	To declare the type of workflows thar a process has to follow.
EuropeanUnionFunding	Project description
Enquiry	Tender question.
Framework	To declare if a process is a Framework
Identifier	Auxiliar entity to declare identifiers
Item	A product or service
JointProcurement	Type of procurement
LegalProceedings	
Lot	Each or a tender lots
LotDetails	
LotGroup	Group for lots
Milestone	
Objective	
Option	
Organization	
OrganizationReference	
ParticipationFee	
Period	Budget or tener period devinition
Person	
PlaceOfPerformance	
Planning	Organization plan with the budget
ProcedureOutsourcing	
PurposeOfNotice	
RecurrentProcurement	Flag for recurrent processes
RelatedNotice	
RelatedPsocess	
Renewal	
Requirement	
Requirement Group	
Requirement Reference	
RequirementResponse	
ReviewProceedings	
Tender	
Transaction	
TreasuryBudgetSource	
Unit	Auxiliar entity to describe the items
Value	Auxiliar entity for economic values
Person	Person declaration form buyers
Value	Auxiliar entity for vaules
ValueBreakdown	
ValueTax	
Variant	

_Enums	List of values

5.10.3 Functional relationship

This component is described in the functional document “EBRD_MTender_Technical Specifications_v16.0.docx”, in section 4.4.

6 Business Process Engine

The business processing engine (BPE) is the logical layer responsible for identifying an event, and then selecting and executing the appropriate reaction. It can also trigger a number of assertions. Processing involves tracking and analyzing streams of data from events to support better insight and decision making. With the recent explosion in data volume and diversity of data sources, this goal can be quite challenging for architects to achieve.

6.1 Architecture

A high-level architecture for building BPE is shown below. Depending on use cases, the components involved may vary.

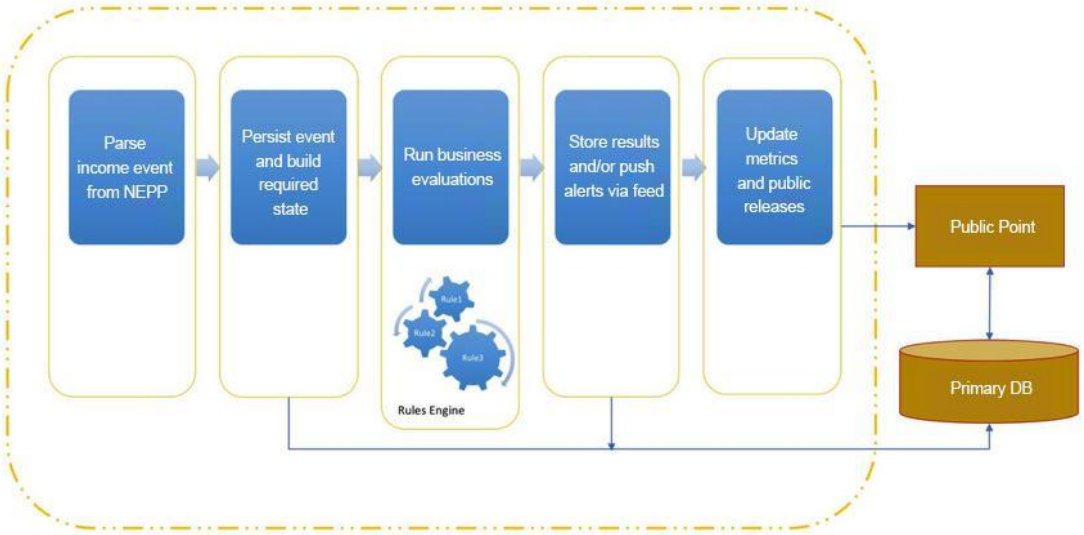


Figure 15. General scheme of the Business Process Engine flow

6.2 Components

The general set of BPE components is as follows:

- *eBudget* - component allows the online definition and preparation of expenditure items, funding sources, periods of budgeting and budgets in structured form.
- *ePlanning* - component allows CAs to construct their procurement plan by scheduling of procedures during defined period as well as identification of potential individual procurement plans to be aggregated in order to launch a repetitive procedure.
- *eAccess* - component is responsible for receiving, validation and saving of core information about the procurement according to selected procurement method, geography, etc and it amendments. Apart from the initiation of a new Contracting Process, the eAccess includes the links to all related processes (such as definition of used budget or conducted prior notices).
- *eClarification* - component provides the option to ask questions and receive answers regarding the conditions of particular procurement as well as allow CAs to answering questions from EOs.
- *eSubmission* - component allows EOs to respond to Contract Notice by submitting their quotations electronically to a CA, including using templates created by the CA. On the CAs side, this component generates a standardized schema for a bid or proposal in the relevant procurement method, stores received bids or proposal until expiry of tender deadlines and allows secure opening of the received tenders upon expiry of the tender deadlines.
- *eQualification* - component handles legal, economic and quality qualification of the tenderers and its proposals. It is responsible for accessing EOs master data. It is also used for shortlist management of procedures that allow shortlisting prior to submission of tenders.
- *eAuction* - component facilitates the configuration and management of reverse auction held electronically.
- *eAwarding* - component allows preparation of the Contract Award Notice in a structured way. It also ensures exchange of documents with the tenderer during the awarding phase. eAwarding is responsible for all actions once evaluation is completed until the contract is created.
- *eContracting* - component allows rearation and creation of the Awarded Contracts in a structured way. It also ensures the interaction with the eGov infrastructure (Treasury, state registers, etc) during Contract preparation phase. eContracting is responsible for all actions once contract is created as a result of evaluation under specific procurement procedure until this contract is implemented.

- *eNotice* - component is in charge of generating Notices using the data available in the eAccess and other components. The notices will be generated in a structured way available to download or to send to third parties. The module will also allow the creation of all other needed types of notices for all types of procedures based on the tender specifications introduced in the eProcurement system.

6.3 Environment

In addition to the BPE and its components, the system also includes several additional logical and functional components:

- Chronograph
- Feed Point
- Master Data Service
- Document Service

6.3.1 Chronograph

As soon as described architecture considered as an event-driven system, there must be a component responsible for storing and executing of scheduled events that may have any influence in any process executed in the system.

Methods

Chronograph component includes a single method responsible for the scheduling of different events under different processed executed in the system:

Schedule.Event()

Methods responsible for scheduling an event to be tracked against general systems' time

```
POST /schedule?ocid=...&action=...&launchTime=...&phase=

{} // payload according to internal request model

201                                                                    OK
Content-Type: application/json; charset=UTF-8
```

Incomes

Name	Type	Description	Obligation
Ocid	string	OCID of the target phase of the contracting process	mandatory
Action	string	schedule | cancel | replace	mandatory

launchTime	date-time	Time when this event is requested	mandatory
Phase	string	tenderPeriod.end | enquiryPeriod.end | awardPeriod.start	mandatory

6.3.2 Feed Point

The component is responsible for arrangement of real-time communication between different internal and external components of the system for transferring a personalized data as a closed feed.

Methods

Feed Point component includes following methods:

Title	Method	Description
Send.Message()	POST	Send a message to specific consumer

6.3.3 Master data Service

Master Data Service component provide master data information based on diferent ISO's. The system provides this data using the following methods:

Methods

Title	Method	Description
Get.Countries()	GET	Method returns list of countries and its codes as in ISO-3166
Get.Regions()	GET	Method returns list of regions and its codes as in ISO-3166
Get.Localities()	GET	Method returns list of localities and its codes as in UN/LOCODE
Get.RegistratoinSchemas()	GET	Method returns list of schemes for specified country as in IATI
Get.Currencies()	GET	Method returns list of currencies and its codes as in ISO-4217
Get.MeasureUnitsGroups()	GET	Method returns list of measure groups as in UN/CEFACT
Get.MeasureUnits()	GET	Method returns list of measure units and its codes as in UN/CEFACT
Get.CPV()	GET	Method returns a list of classes specified level of CPV
Get.CPVS()	GET	Method returns a list of classes of CPVs
Get.PMD()	GET	Method returns a list of codes of supported procurement methods

Get.Languages()	GET	Method returns list of languages and its codes as in ISO-639

6.3.4 Document Service

'Document Service' is a System Environment Component responsible for registering, uploading, archiving, organising and control of access to all documents (files), enabling long-term preservation of them in digital format and ensuring that they can be easily retrieved without conversions. Service also provides basic document management functionality for receipt, dispatch, storage and retrieval of documents.

Requirements

Supported Document Formats (Media Types)

According to used Media Types Data standard next formats are supported by Document Service:

Images: jpg, png, gif, tif

Documents: doc, docx, xls, xlsx, pdf, rtf

Archives: rar, zip, 7s

Components

As a single part of System environment Document Service consist of several components:

File Repository Management System - Document Service Gateway

Gateway managing the access to the file repository: controls access and reads the files from the repository.

File Repository - Storage

Storage of the files attached to the tender notices, offers, and complaints.

Methods

Document Service component includes following methods:

Title	Method	Description
Register.Document()	POST	Method for registration of the new document
Upload.Document()	POST	Method to load a body of the document
Publish.Document()	POST	Method to publish a document

Get.Document()	GET	Method to retrieve a document

Registered.Document()

Register a document that will be nested in some information entity and a backup location for uploading the document file to the file repository. To register any document hash of such document should be send to DS. The result of document registration is the permanent link (URL) of such document in the file repository that will be returned within respond of DS as part of success data-package. This link can be put to parent entity an attribute of Document right after success registration. Further, a registered document with an identical hash must be uploaded.

Flow description

When document is received to be registered DS performs following actions:

1. The file size is checked - if the value is exceeded (50 Mb), the process is thrown
2. The file format is checked for a valid format
3. In case of a valid format, the file is saved:
   a. description is stored in the database according to the model
   b. Returns file descriptions to the client according to the attribute composition

Example

```
POST                                    /registration                                    HTTP/1.1
Authorization:
Content-Length:                                                                                58
Content-Type:                                                                      application/json
Host: HOST

{
  "data":                                                                                      {
    "hash":                                                  "9a0364b9e99bb480dd25e1f0284c8555",
    "weight":                                                                           "1024",
    "fileName":                                                                      "file.txt"
  }
}

201                                                                                    Created
Content-Type: application/json; charset=UTF-8

{
  "data":                                                                                      {
    "id":                                                    "9a0364b9e99bb480dd25e1f0284c8555",
    "url":                                  "https://HOST/get/389684cc28c242b79c97c56be5142e25",
    "dateModified":                                                   "2017-12-25T17:05:56.044677"
  }
}
```

Upload.Document()

Previously registered Document could be uploaded into File Repository (Storage) latter. There is a time-frame setting that limits such “letter” but this setting defines by unique value in every particular implementation of eProcurement.Systems toolkit.

Flow description

When document is received to be uploaded DS performs following actions:

1. The owner is checked for compliance with the one to whom the document registration url was issued
2. The hash is calculated using the checksum algorithm
3. Compare the checksum of the received file and stored in the reserve by reference. In case of mismatch, an exception is returned and the process is terminated.
4. It is checked for a valid format
5. In case of a valid format, the file is saved:

Example

```
POST                          /upload/389684cc28c242b79c97c56be5142e25                        HTTP/1.0
Authorization:
Content-Length:                                                                                       58
Content-Type:              multipart/form-data;             boundary=----------a_BoUnDaRy572732436472$
Host:                                                                    storage.eprocurement.systems

------------a_BoUnDaRy572732436472$
content
------------a_BoUnDaRy572732436472$--

201                                                                                           Created
Content-Type: application/json; charset=UTF-8

{
  "data":                                                                                             {
    "url":                                    "https://HOST/get/389684cc28c242b79c97c56be5142e25"
  }
}
```

Get.Document()

If document is registered and stored in the Document Service it can be retrieved by any user after publication date for specific document is achieved (document is public and relevant link is published via Public Point). To retrieve needed file simple GET-request should be send to the Document Service.

Example

```
GET                           /get/389684cc28c242b79c97c56be5142e25                          HTTP/1.0
Host:                                                    storage.eprocurement.systems

200                                                                                             OK
Content-Disposition:                        attachment;                        filename=file.txt
```

```
Content-Type: text/plain; charset=UTF-8
```

6.4 External Environment (integrations)

The MTender system integrates the connections with the following government tools in order to manage the services and information.

- Users' Authentication
- Notification Service
- Digital signature
- Logging service
- Treasury
- Payment system

6.5 Sequence diagrams

This section describes the technical specifications of each of the processes that are managed through the Business Process Engine functionality, using accurate sequence diagrams.

Functionality	Description
001. BPE-EI	Management of 'Expenditure Item'
002. BPE-FS	Management of 'Funding Source'
003. BPE-PN/PIN	Management of 'Periodic Notice' or 'Prior Information Notice'
004. BPE-CN	Management of 'Contract Notice'
005. BPE-ENQUIRY	Management of clarification
006. BPE-PS/PQ-BID	Management of submission (pre-selection/pre-qualification)
007. 007.BPE-PS/PQ-TENDERPERIOD:END	Management of deadline of the submission stage
008.BPE-PS/PQ-AWARDING:STEP	Management of awarding (pre-selection/pre-qualification)
009.BPE-PS/PQ-AWARDING:PROTOCOL	Management of end of awarding process
010.BPE-PS/PQ-AWARDING:END	Management of end of awarding stage and all phase
011.BPE-PS/PQ-START-NEW-STAGE	Management of start of next phase
012.BPE-EVALUATION-TENDERPERIOD:END	Management of end of bidding stage (evaluation)
013.BPE-EVALUATION-AWARDING:STEP	Management of awarding stage (evaluation)
014.BPE-EVALUATION-AWARDING:PROTOCOL	Management of end of awarding process

015.BPE-EVALUATION-AWARDING:END	Management of end of awarding stage and all phase
016.BPE-CONTRACTING-CONTRACT:PREPARATION	Management of contract preparation
017.BPE-CONTRACTING-CONTRACT:ACTIVATION	Management of closing of evaluation

BPE-EI: Management of 'Expenditure Item'

Create new EI

Incomes

payload and params	`EI command model according to API documentation`
api endpoint	`/do/ei`

Execution diagram

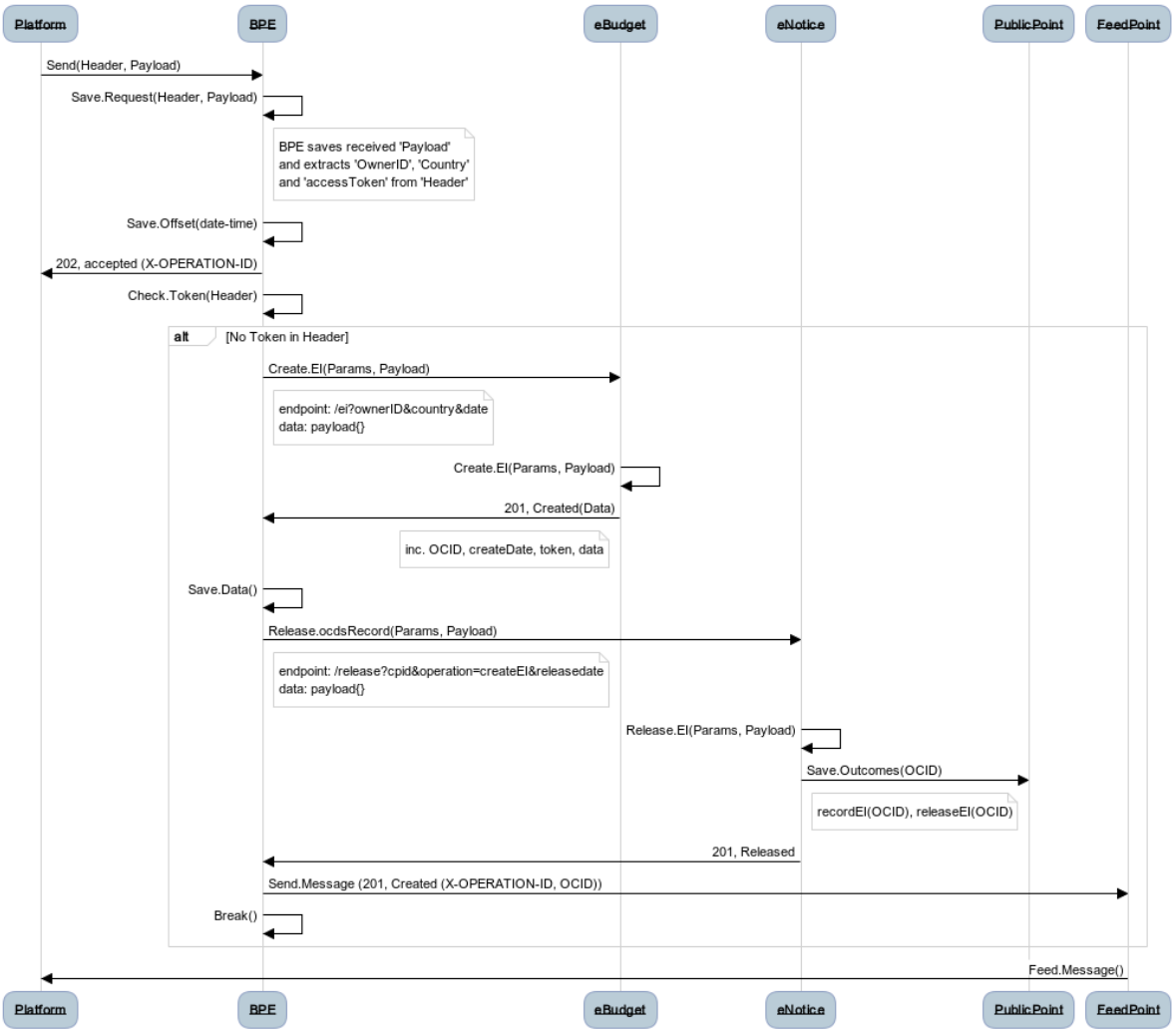


Figure 16. Process sequence diagram (https://goo.gl/MC13EM)

001.1.2. Update existing EI

Incomes

payload and params	`EI update model according to API documentation`
endpoint	`/do/ei`

Execution diagram

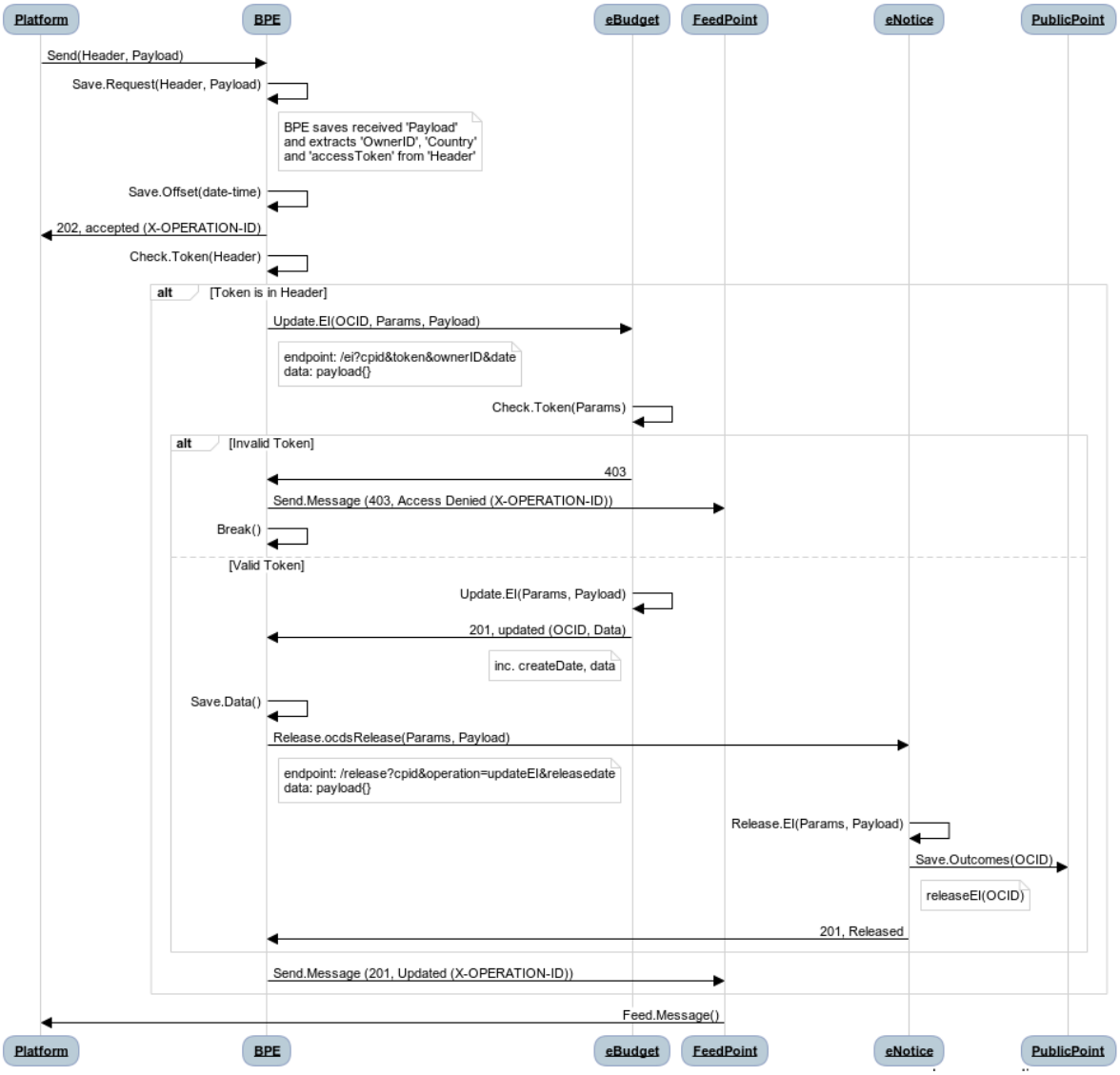


Figure 17. Process sequence diagram (https://goo.gl/Lntvbs)

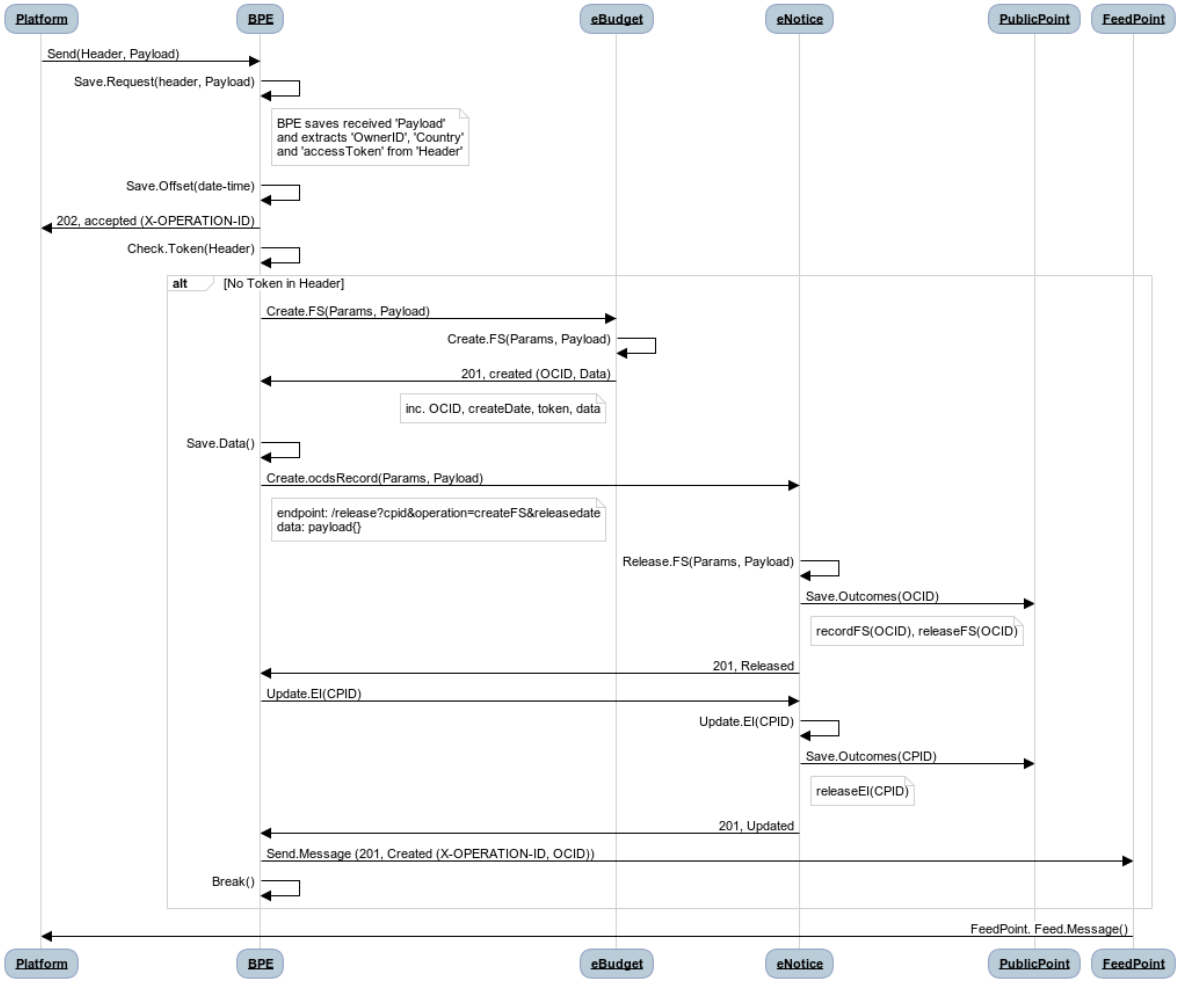
BPE-FS: Management of 'Funding Source'

Create new FS

Incomes

payload and params	`FS command model according to API documentation`
endpoint	`/do/fs`

Execution diagram



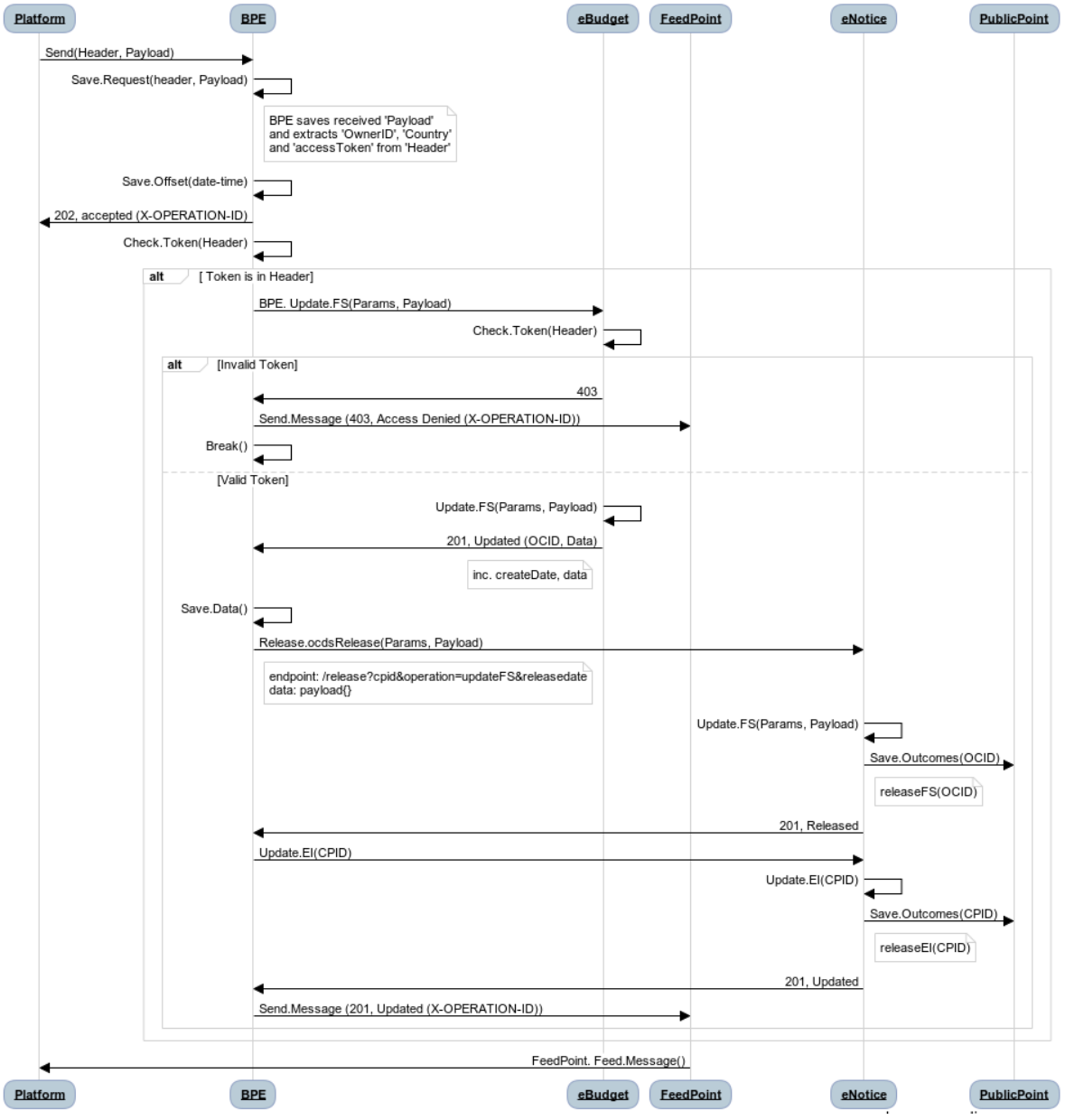
Process sequence diagram (https://goo.gl/Yow9kL)

Update existing FS

Incomes

payload and params	`FS update model according to API documentation`
endpoint	`/do/fs`

Execution diagram



Process sequence diagram (https://goo.gl/yCYg3L)

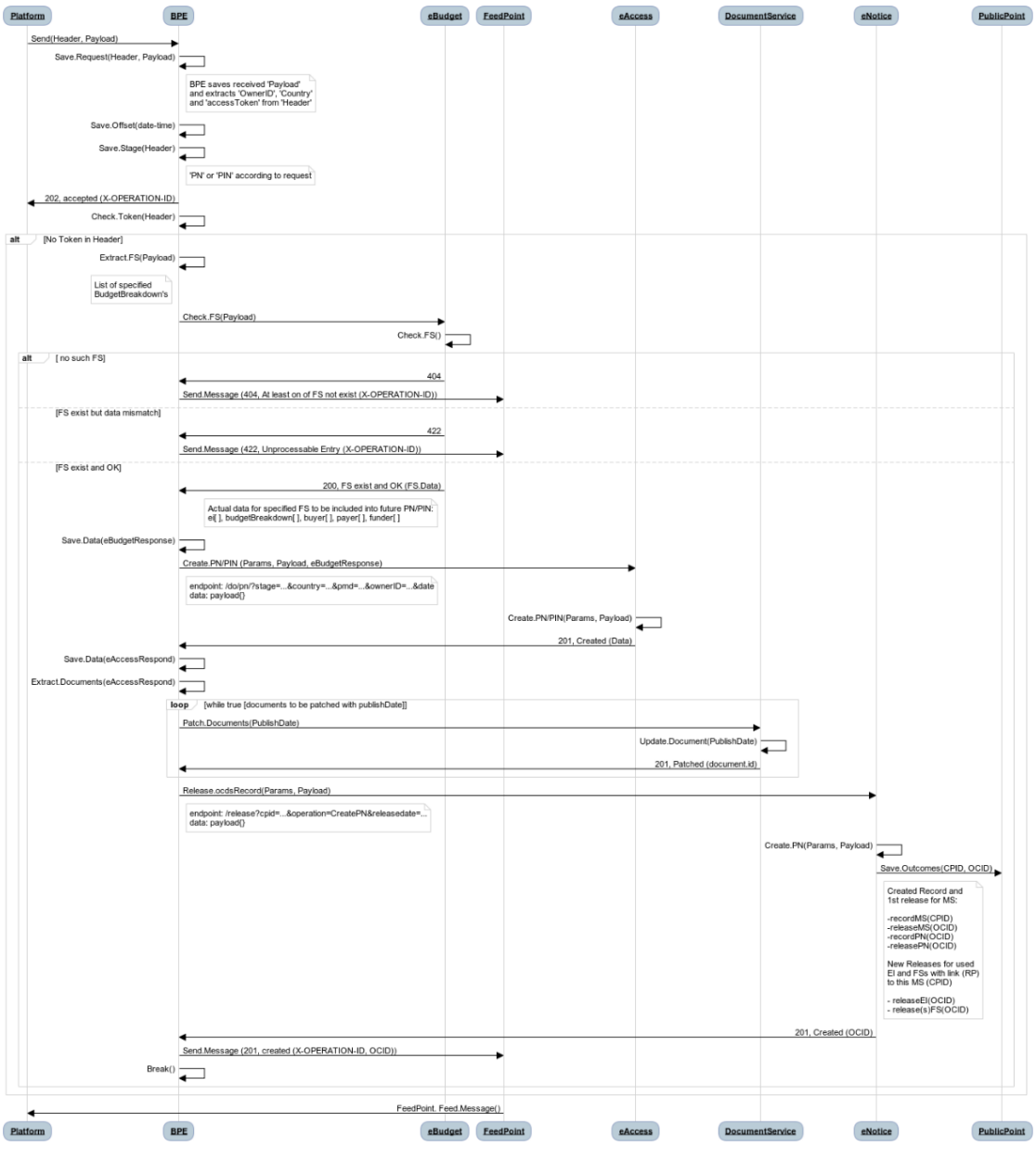
BPE-PN/PIN: creation and management of 'Periodic Notice' or 'Prior Information Notice'

Create new PN/PIN

Incomes

payload and params	`CN command model according to API documentation`
endpoint	`/do/pn(pin)`

Execution diagram



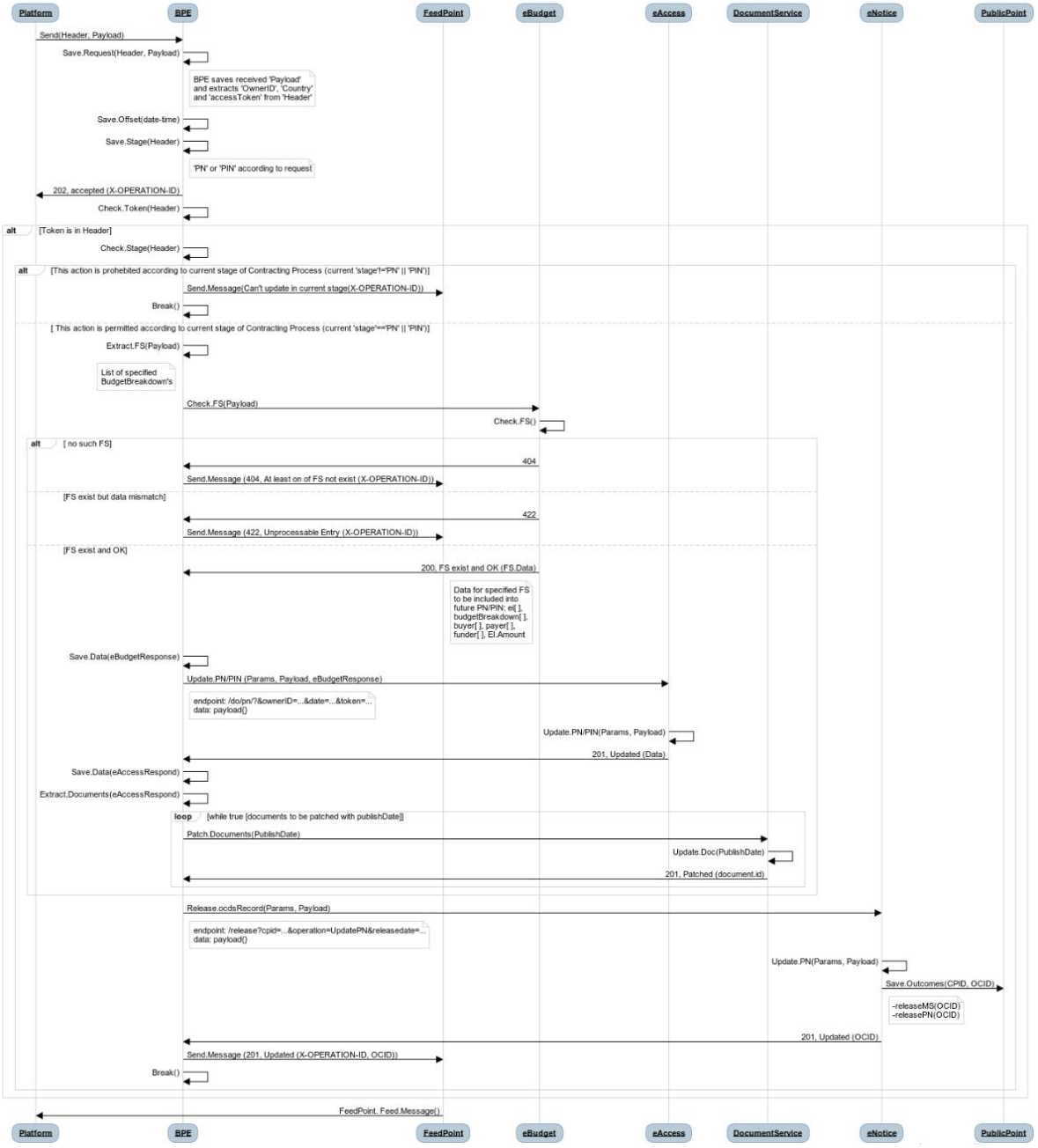
Process sequence diagram (https://goo.gl/bzqJvR)

Update existing PN/PIN

Incomes

payload and params	`CN update model according to API documentation`
endpoint	`/do/pn(pin)`

Execution diagram



Process sequence diagram (https://goo.gl/9HLmvB)

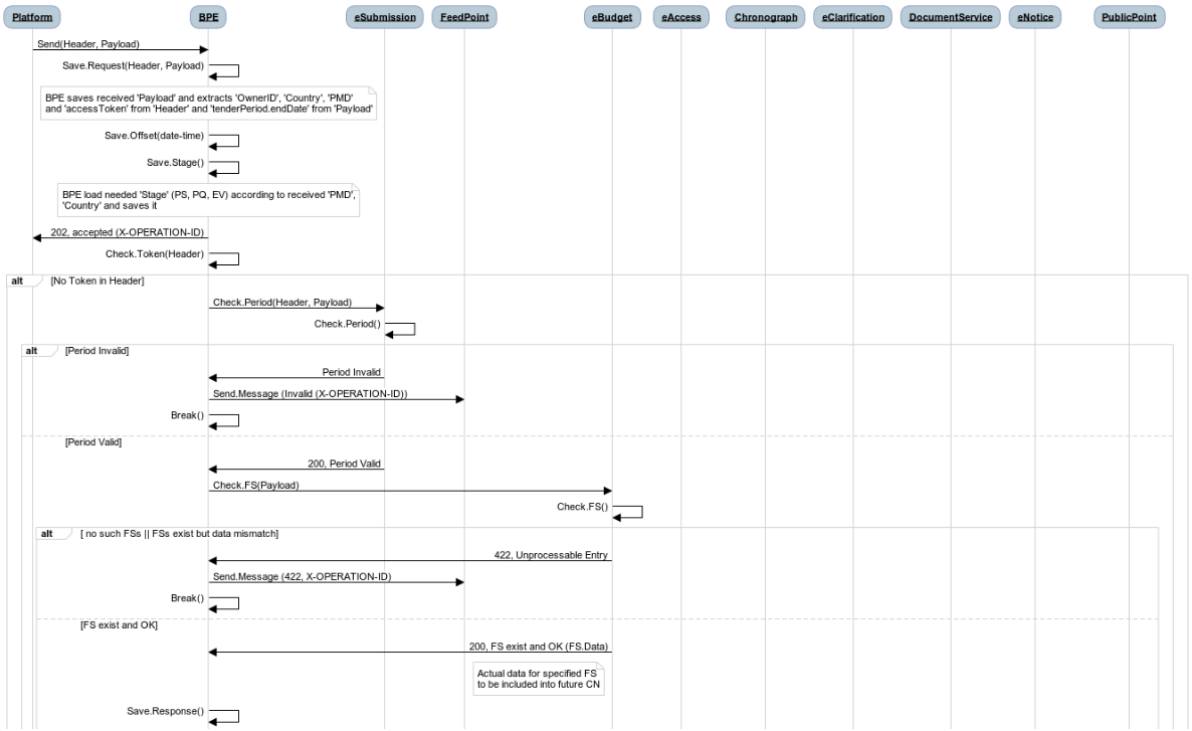
BPE-CN: creation and management of 'Contract Notice'

Create new CN

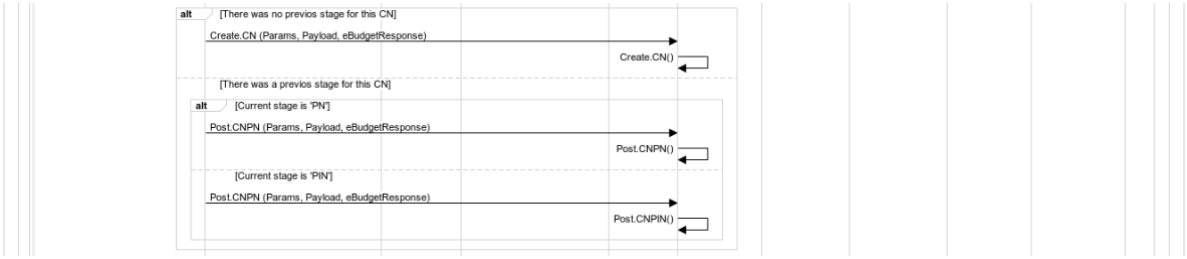
Incomes

payload and params	`CN command model according to API documentation`
endpoint	`/do/cn`

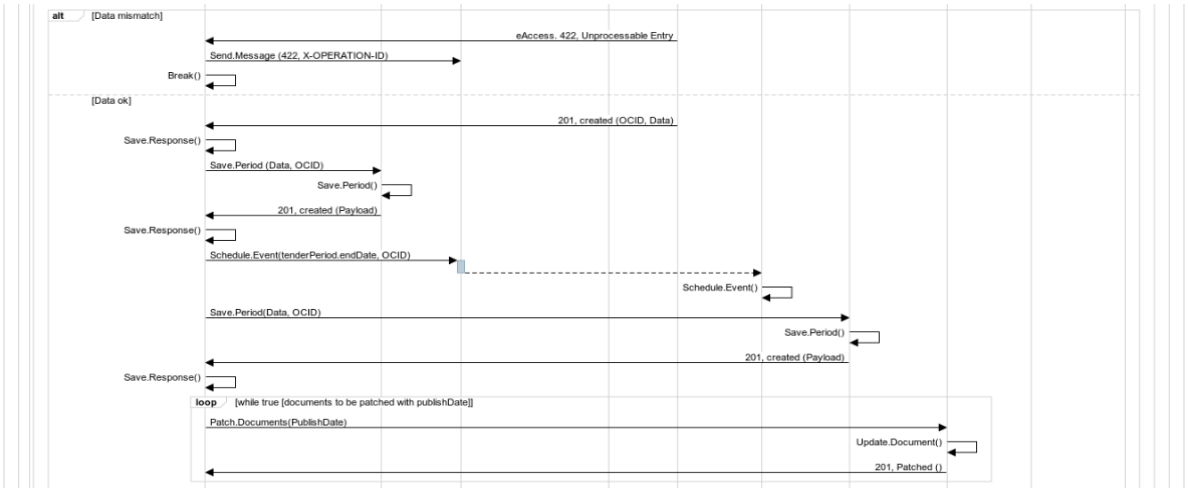
Execution diagram



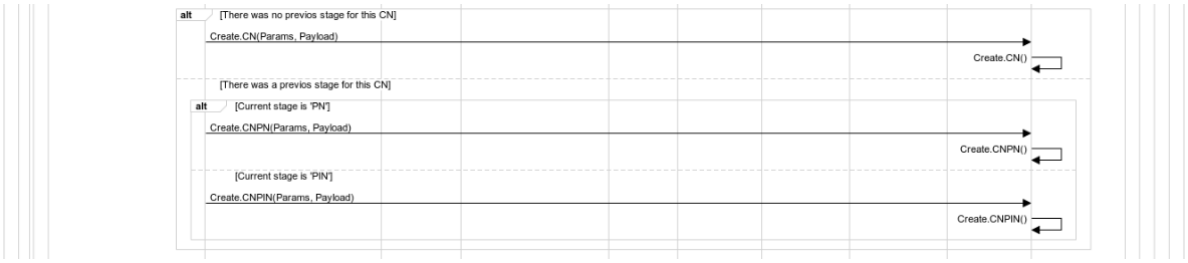
Depending of current stage of Contracting Process different ways of creation of CN will be used



When process will goes forward



Depending of current stage of Contracting Process different ways of publication will be use



When process will goes to end

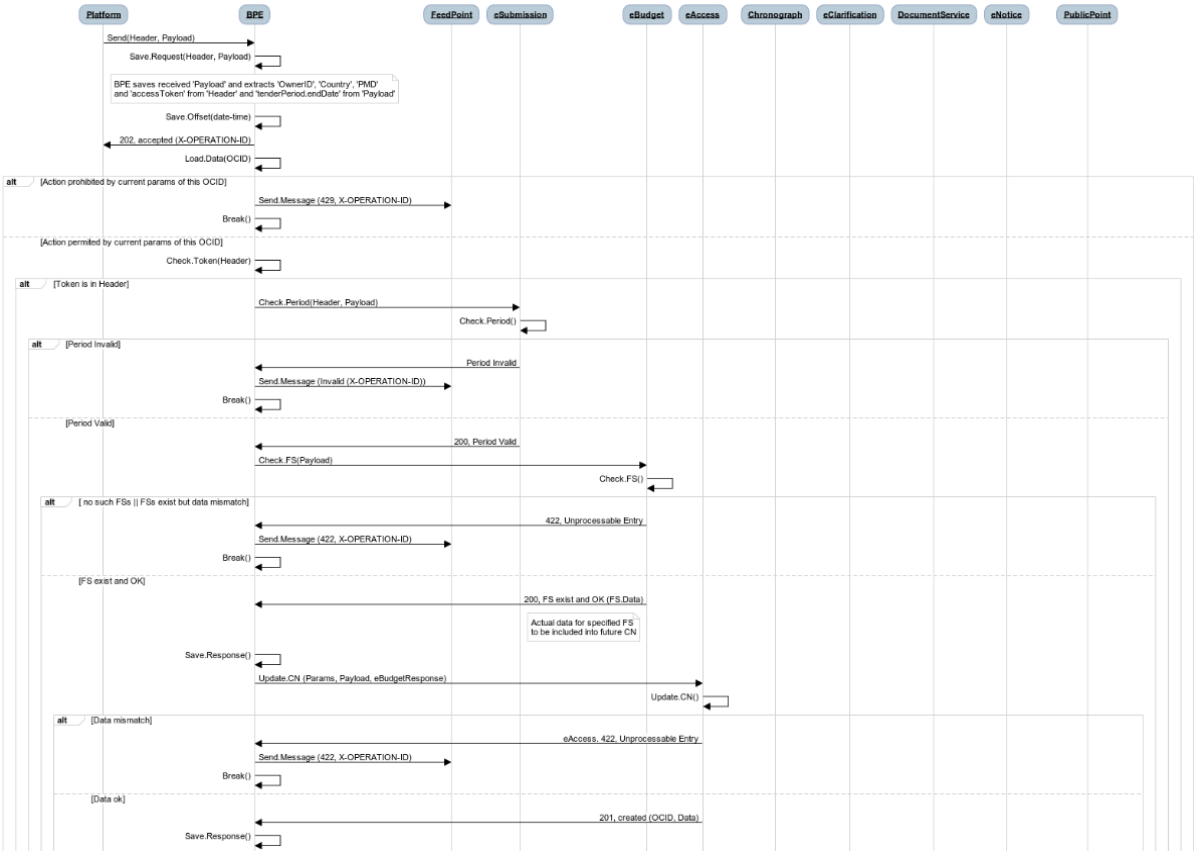
Process sequence diagram (https://goo.gl/K9u2qm)

Update existing CN

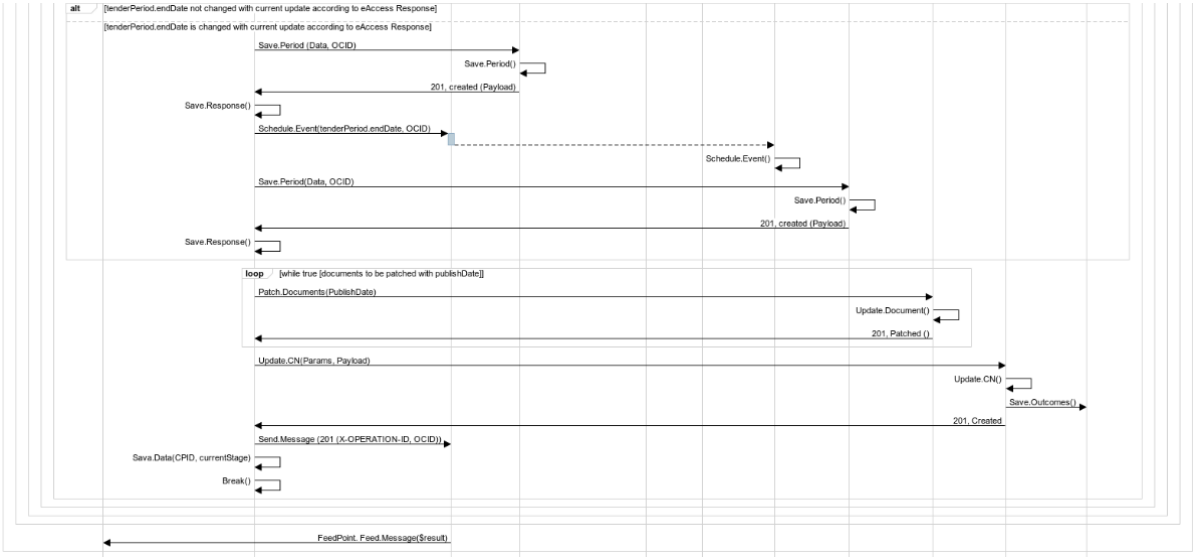
Incomes

payload and params	`CN update model according to API documentation`
endpoint	`/do/pn(pin)`

Execution diagram



Depending of current stage of Contracting Process different ways of publication will be used



Process sequence diagram (https://goo.gl/xCBeXh)

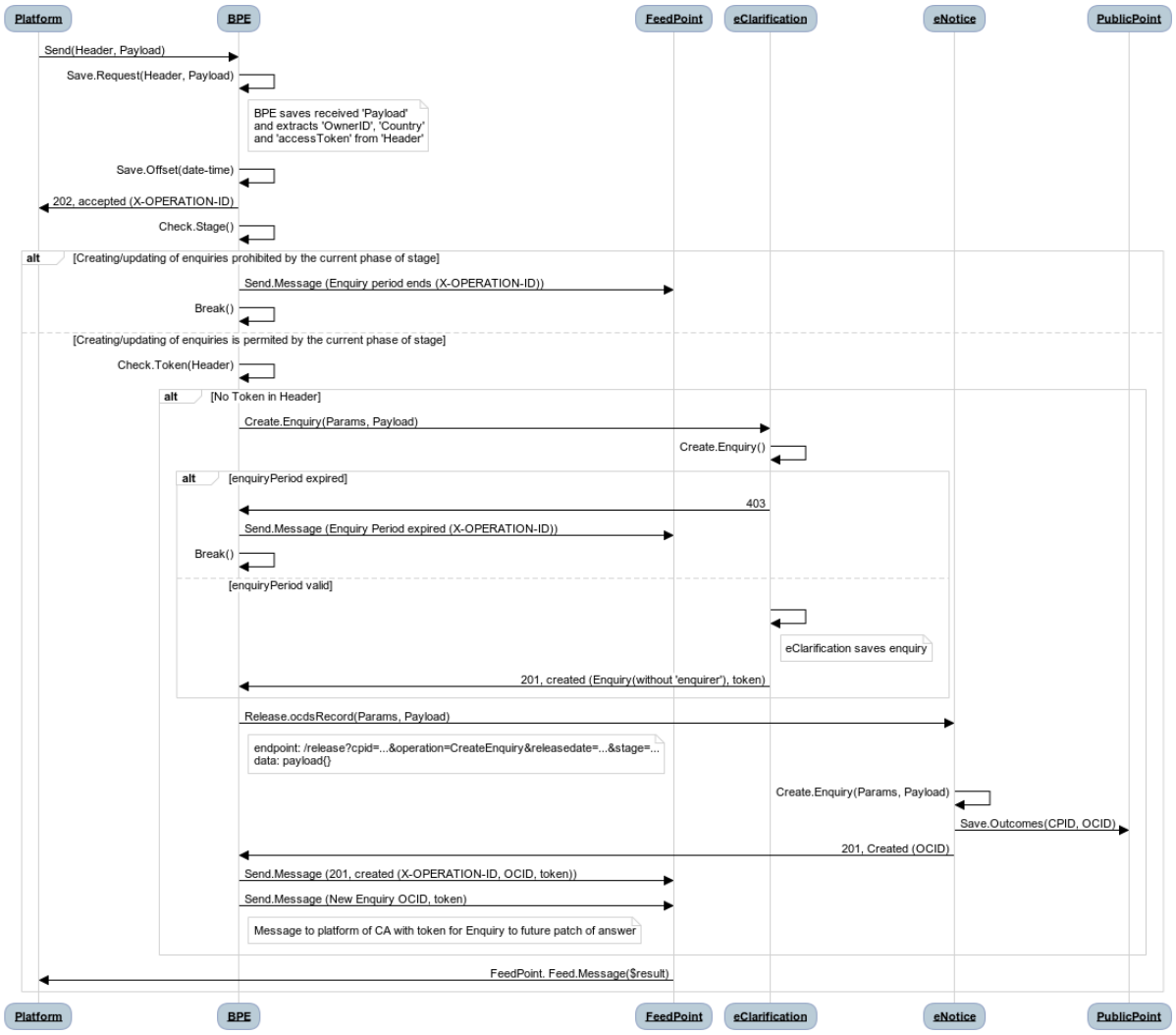
BPE-ENQUIRY: execution of flows of clarification

Create new Enquiry

Incomes

payload and params	`ENQUIRY command model according to API documentation`
endpoint	`/do/enquiry`

Execution diagram



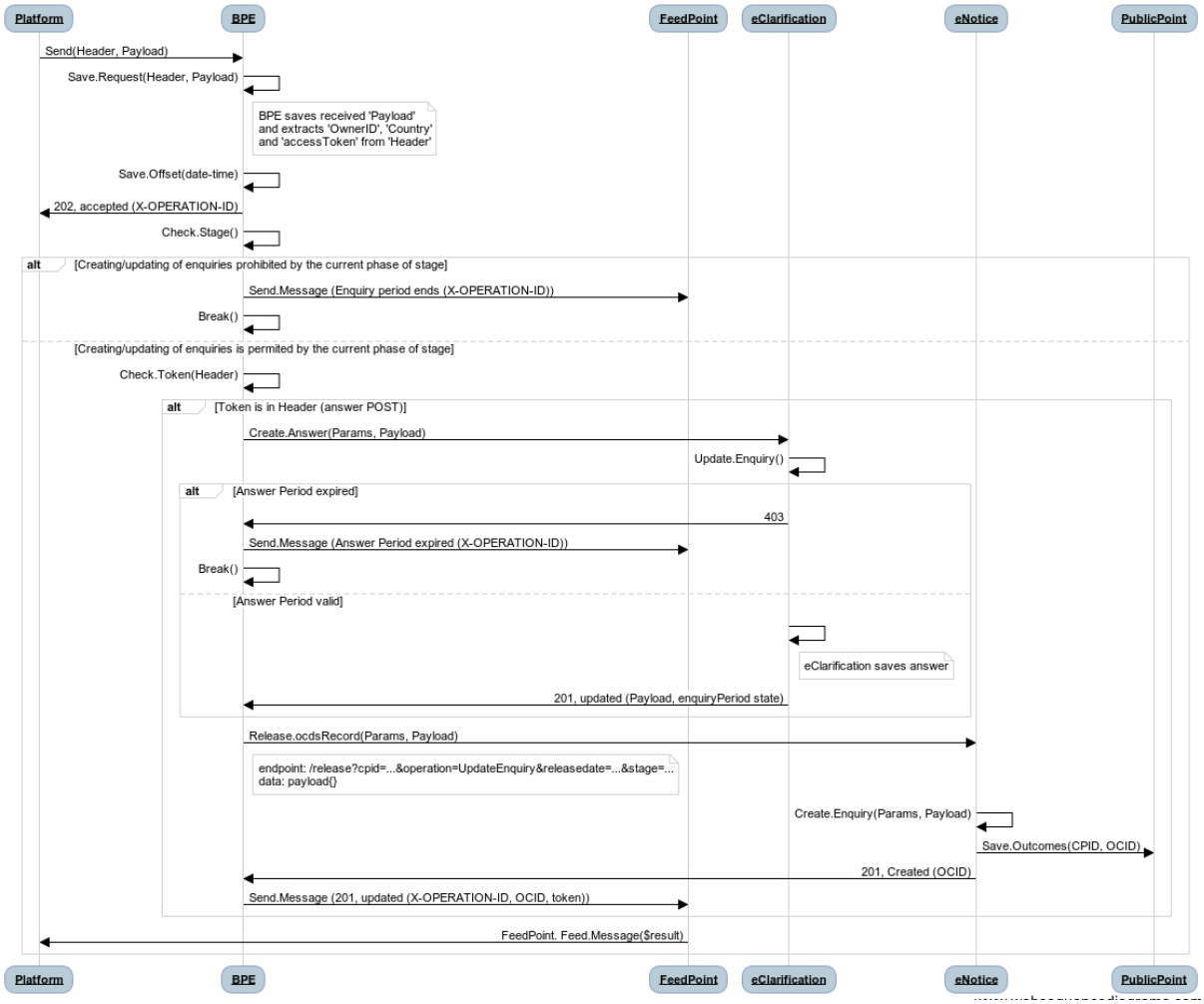
Process sequence diagram (https://goo.gl/b4d4Nv)

Answer on existing enquiry

Incomes

payload and params	`ANSWER command model according to API documentation`
endpoint	`/do/enquiry`

Execution diagram



Process sequence diagram (https://goo.gl/bSh6Lj)

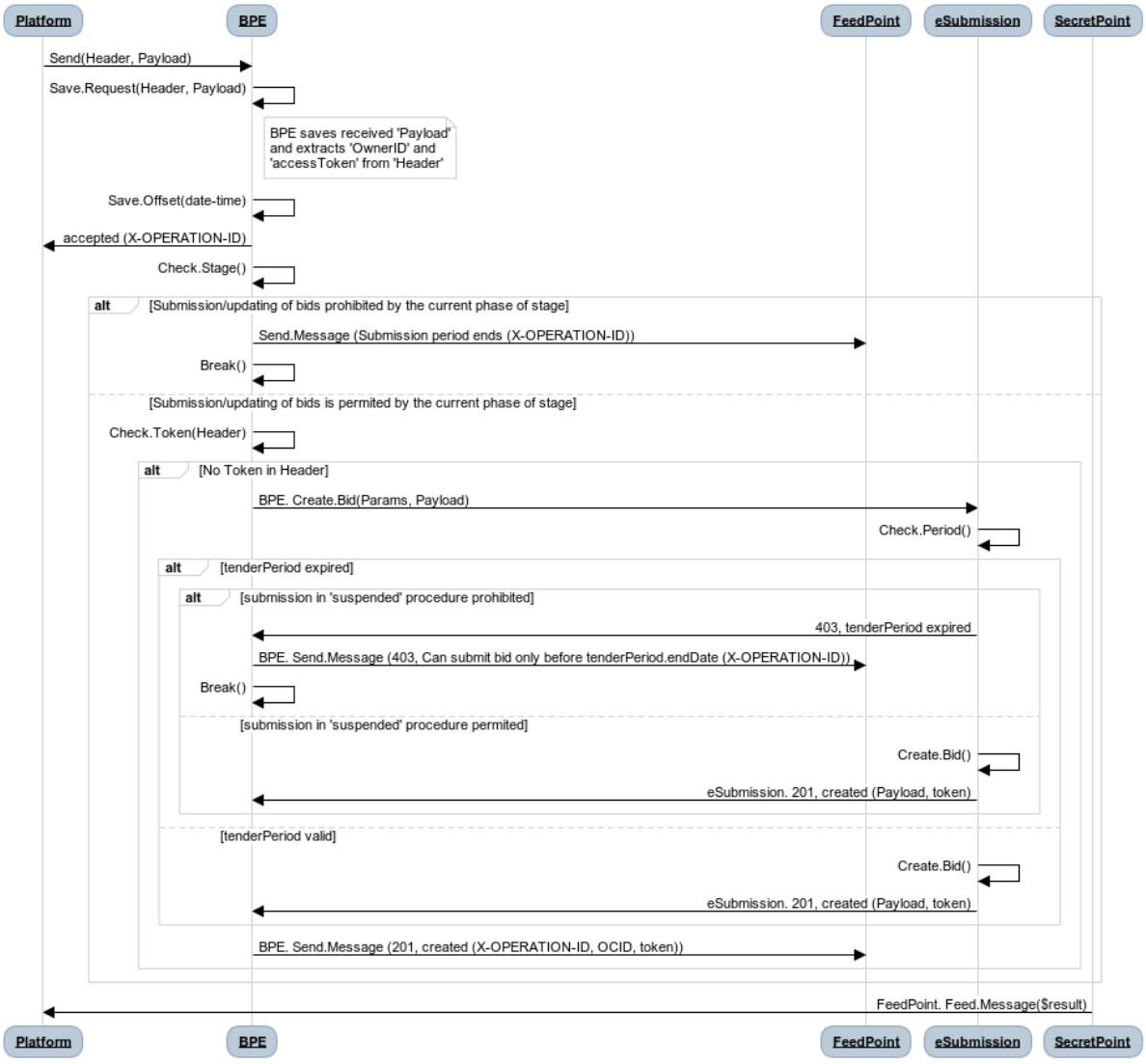
BPE-PS/PQ-BID: execution of flows of submission stage (pre-selection/pre-qualification phases)

Submit new bid

Incomes

payload and params	`BID command model according to API documentation`
endpoint	`/do/bid`

Execution diagram



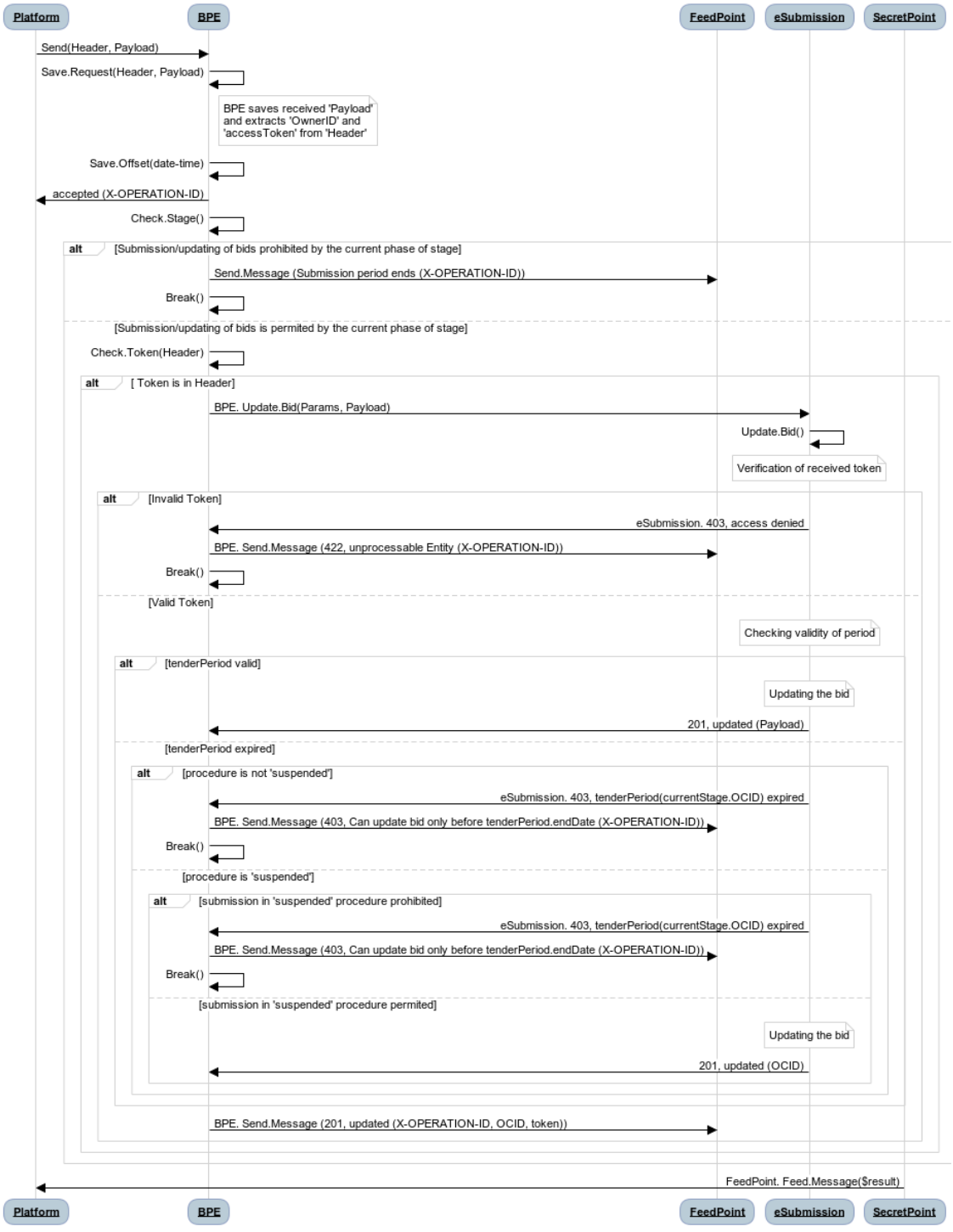
Process sequence diagram (https://goo.gl/h1XXsM)

Update existing bid

Incomes

payload and params	`BID update model according to API documentation`
endpoint	`/do/bid`

Execution diagram



Process sequence diagram (https://goo.gl/HhgmhD)

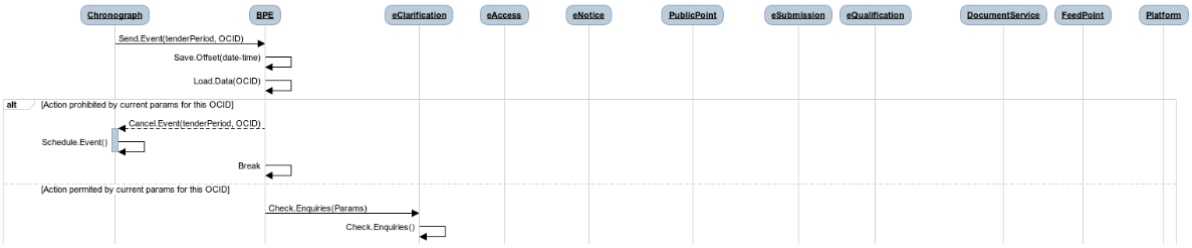
BPE-PS/PQ-TENDERPERIOD:END: management of end of submission stage

Close of submission period in current phase (automated)

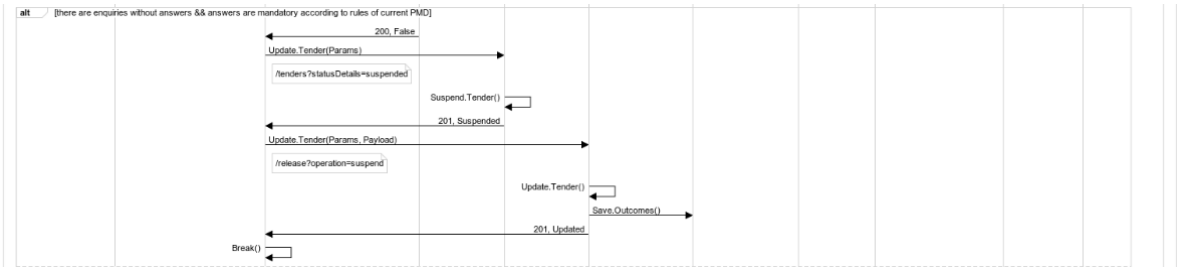
Incomes

payload and params	`automated internal process`

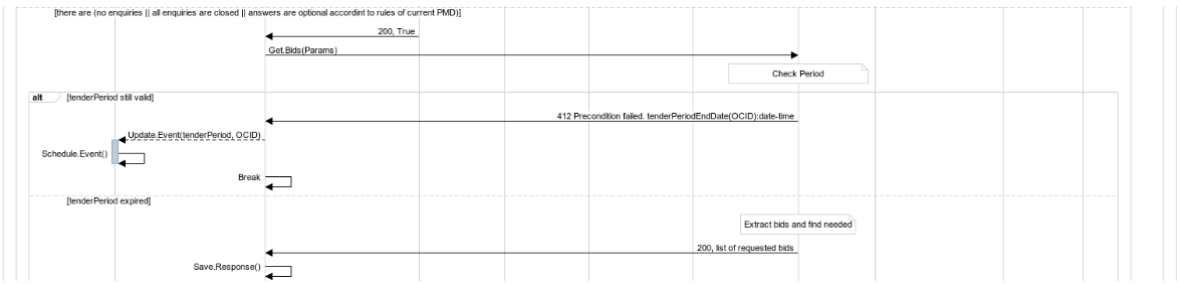
Execution diagram



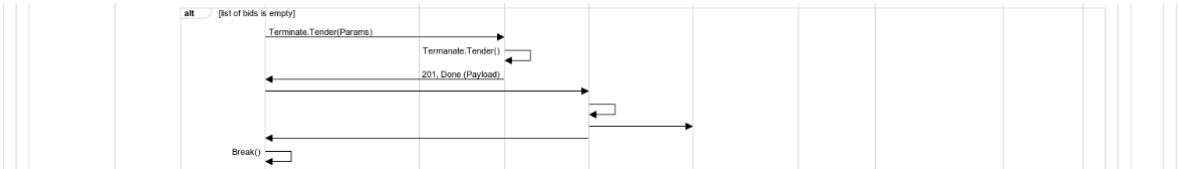
If there are enquiries without answers and procurement method required such answers mandatory, procedure will be suspended (flow will stops) till all needed answers received



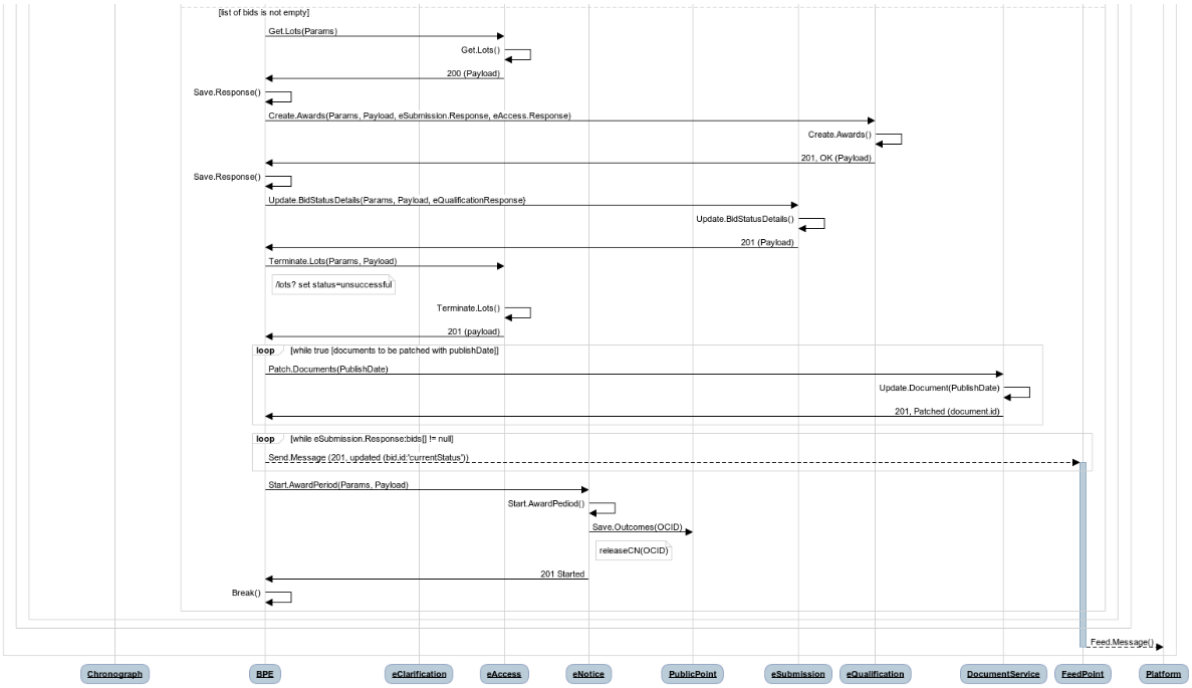
If there are no enquiries without answers or such answers are optional according to current procurement method, process will go to next check



Depending of consistency of array of extracted bids, process may follow in a two different ways: if an array of extracted bids is empty, all contracting process will go to termination as 'unsuccessful'



If an array of extracted bids is not empty, process will go to next check



Process sequence diagram (https://goo.gl/fhyYbW)

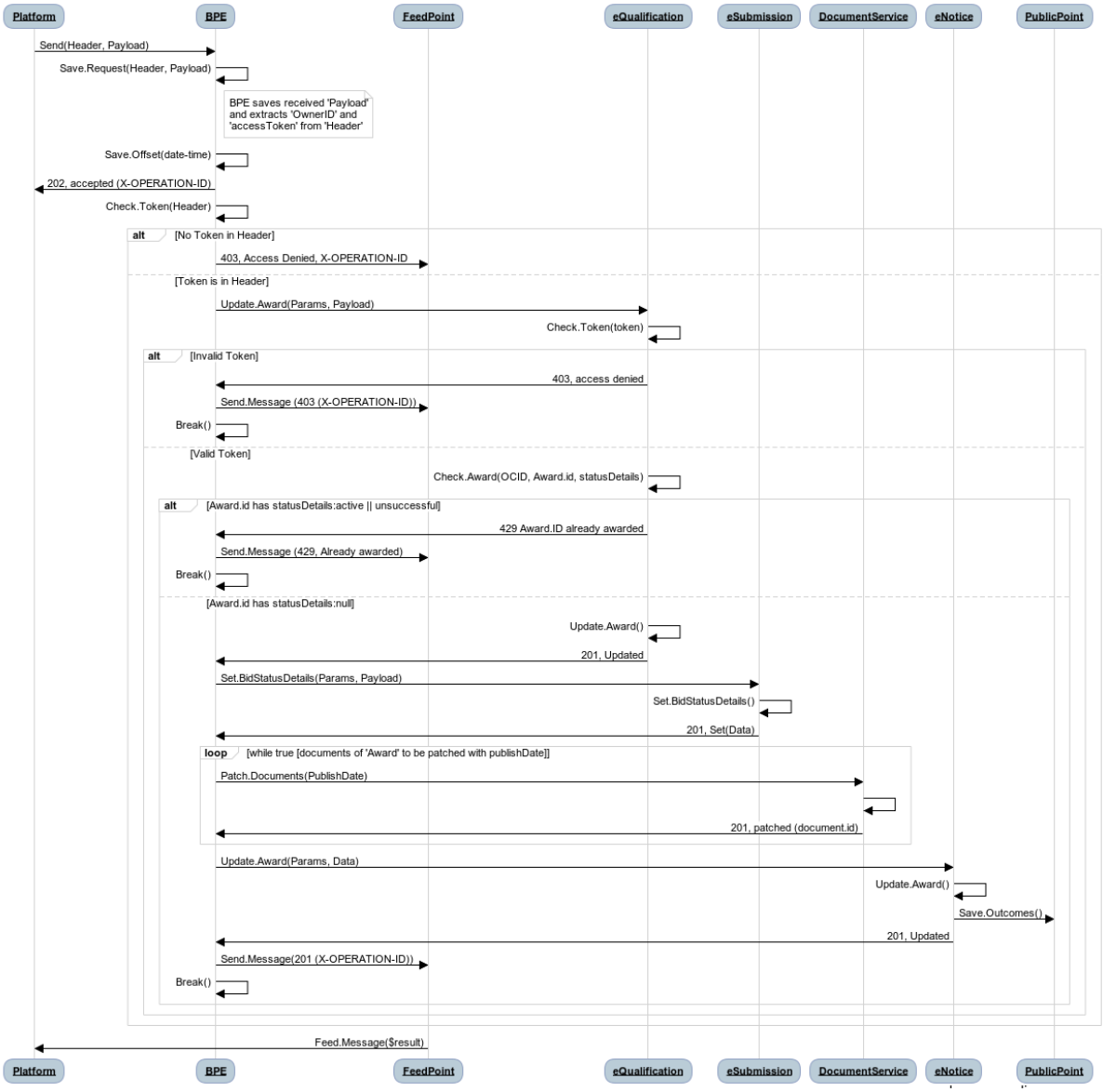
BPE-PS/PQ-AWARDING:STEP: execution of awarding stage (pre-selection/pre-qualification phases)

Update auto-generated 'Award'

Incomes

payload and params	`AWARD update model according to API documentation`
endpoint	`/do/award`

Execution diagram



Process sequence diagram (https://goo.gl/CTBPSD)

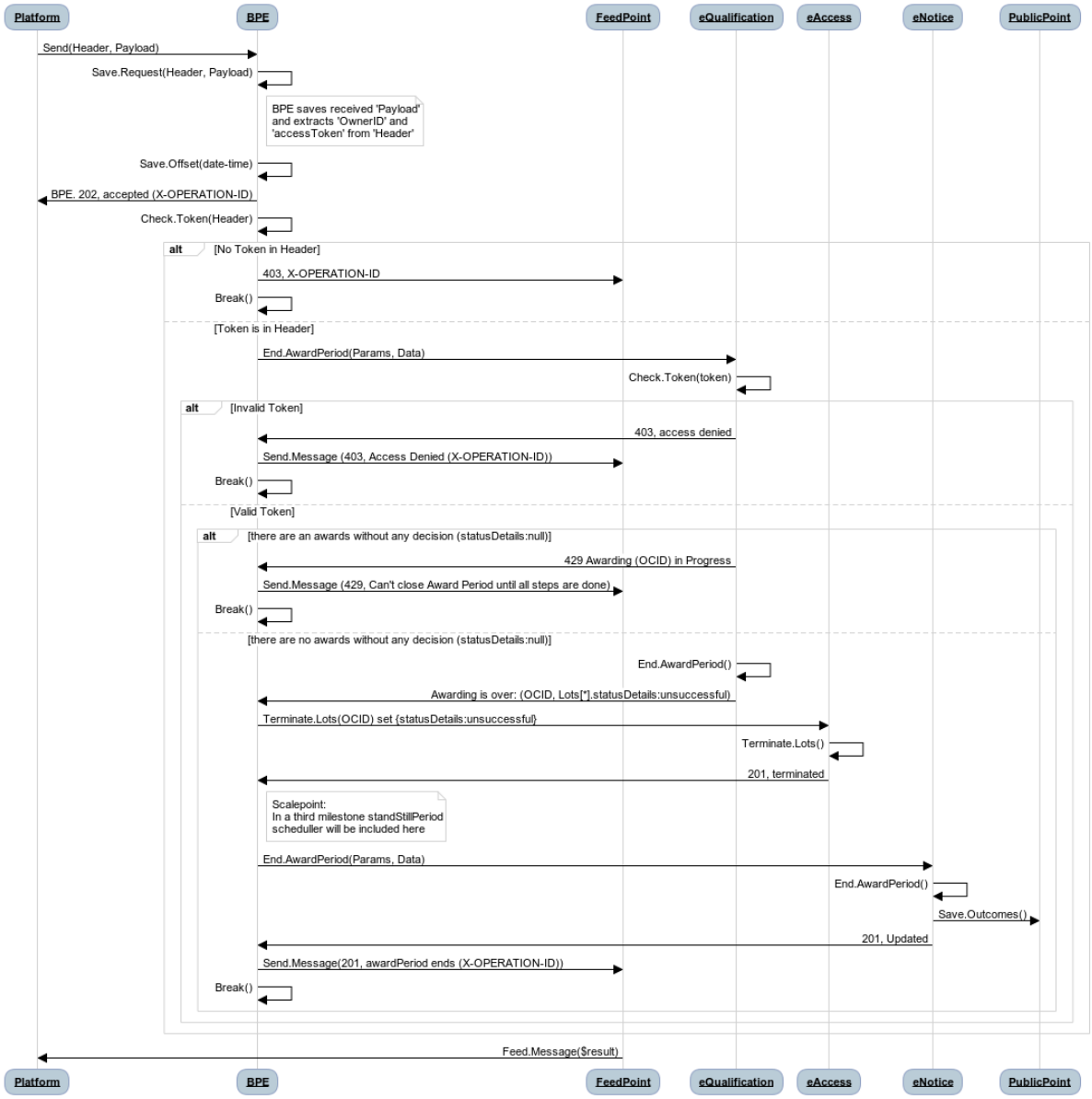
BPE-PS/PQ-AWARDING:PROTOCOL: execution of end of awarding process

Submit 'Awarding Protocol'

Incomes

payload and params	`PROTOCOL command model according to API documentation`
endpoint	`/do/protocol`

Execution diagram



Process sequence diagram (https://goo.gl/HFxsL9)

BPE-PS/PQ-AWARDING:END: management of end of awarding stage and all phase

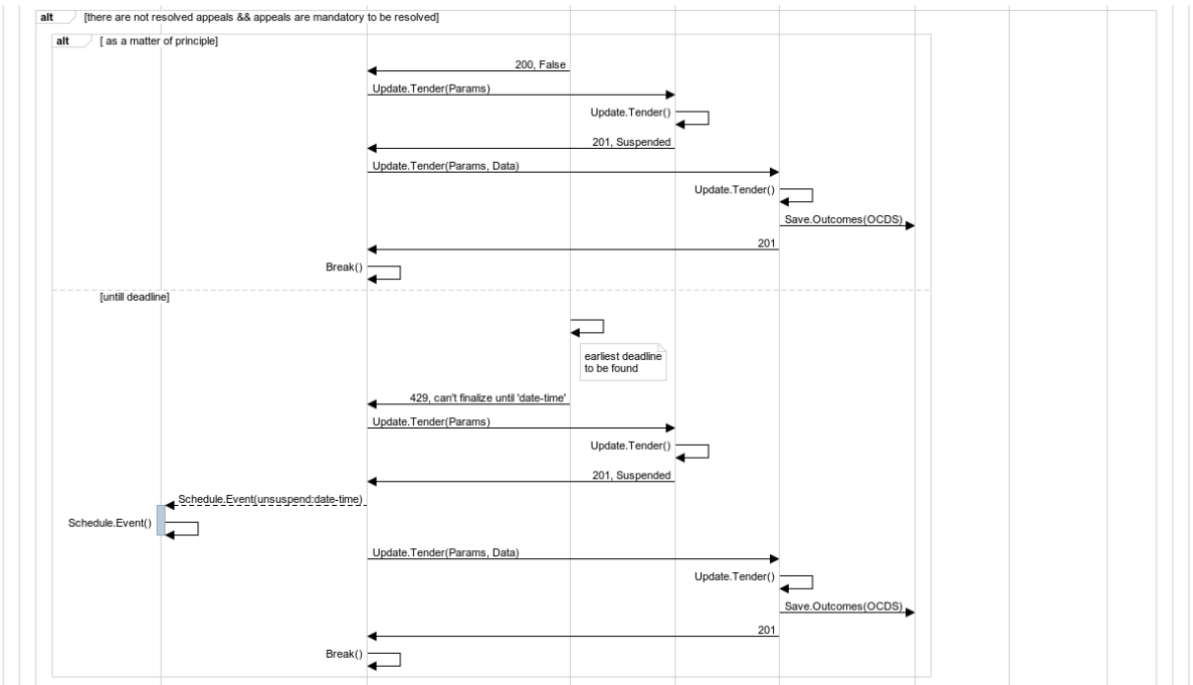
Close of awarding period in current phase (automated)

Incomes

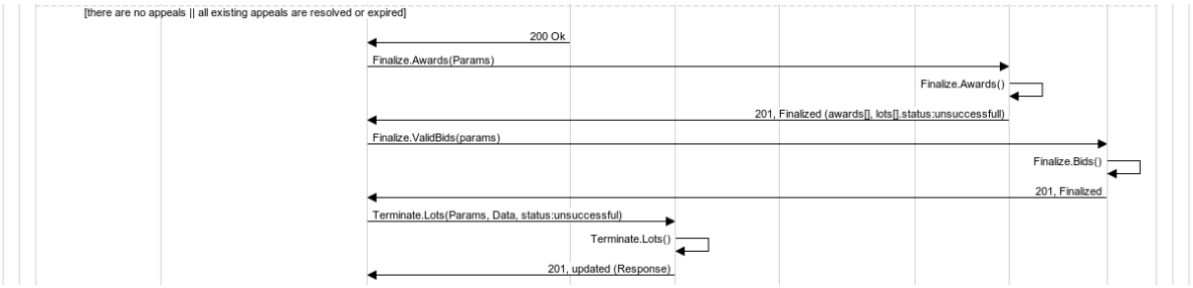
payload and params	`automated internal process`

Execution diagram

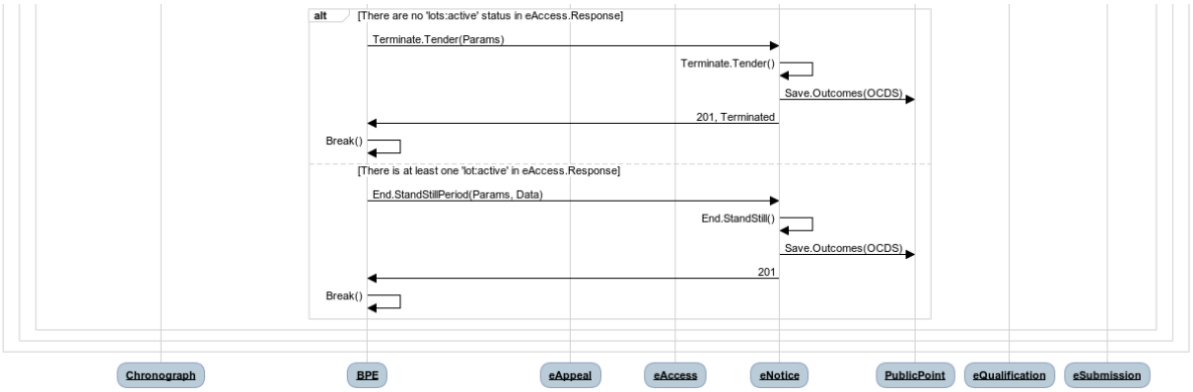
If there are enquiries without answers and procurement method required such answers mandatory, procedure will be suspended (flow will stops) till all needed answers received



If there are enquiries without answers and procurement method required such answers mandatory, procedure will be suspended (flow will stops) till all needed answers received



If there are enquiries without answers and procurement method required such answers mandatory, procedure will be suspended (flow will stops) till all needed answers received



Process sequence diagram (https://goo.gl/LZJWun)

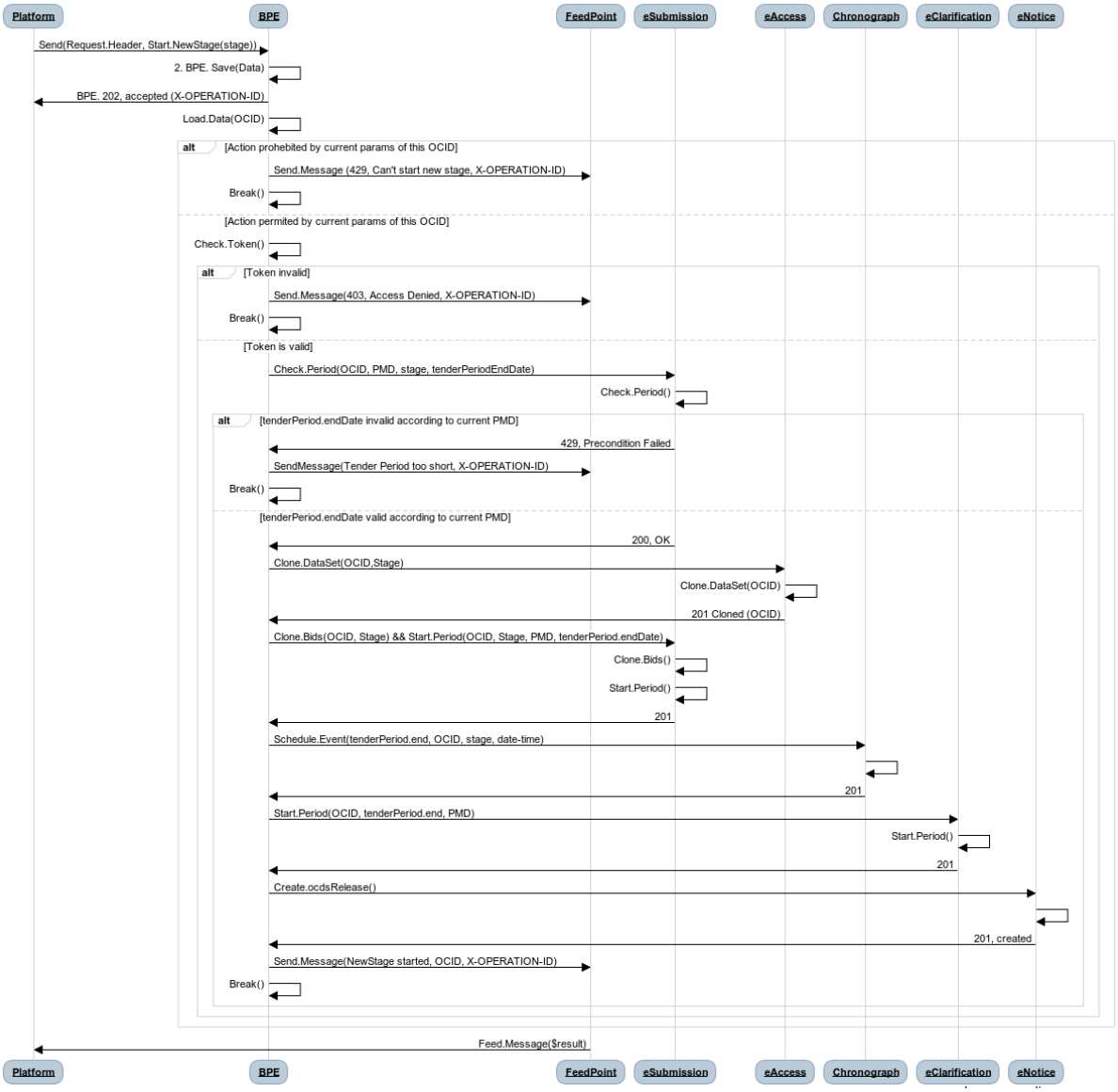
BPE-PS/PQ-START-NEW-STAGE: management of start of next phase

Launch new phase (automated)

Incomes

payload and params	`automated internal process`

Execution diagram



Process sequence diagram (https://goo.gl/3NFUy4)

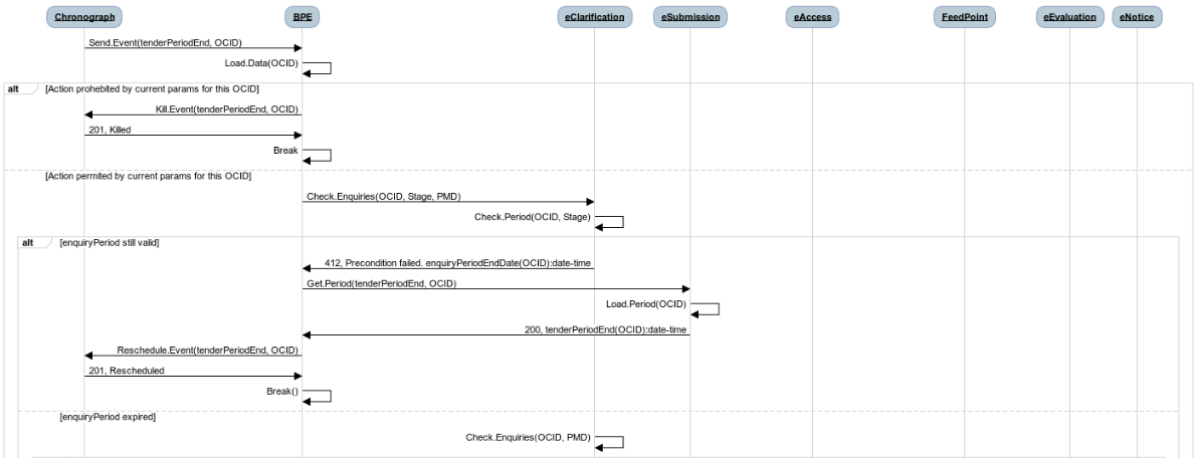
BPE-EVALUATION-TENDERPERIOD:END: management of end of bidding stage (evaluation phase)

Close of submission period in current phase (automated)

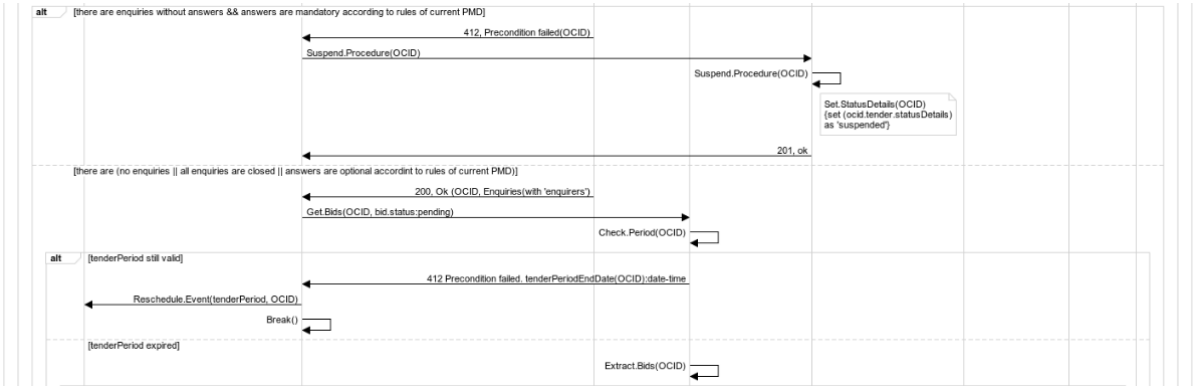
Incomes

payload and params	`automated internal process`

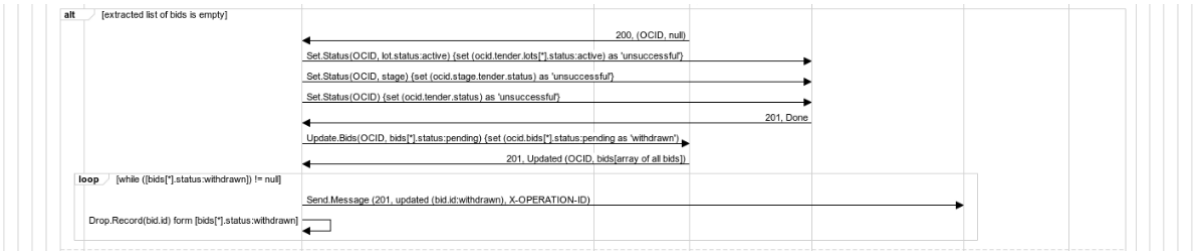
Execution diagram



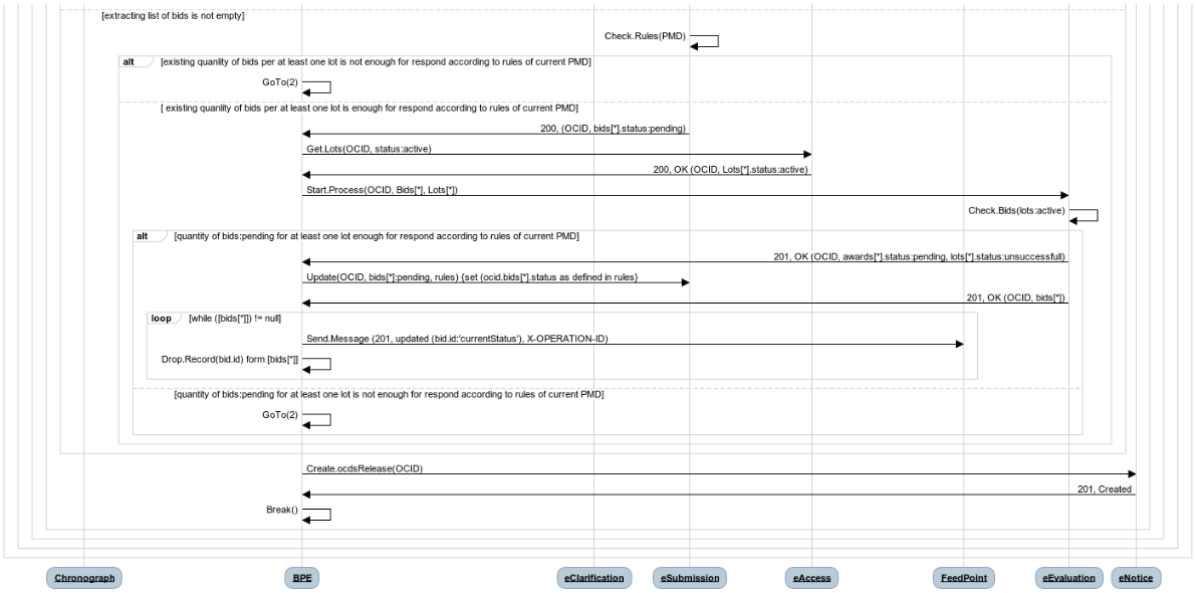
If there are enquiries without answers and procurement method required such answers mandatory, procedure will be suspended (flow will stops) till all needed answers received



Depending of consistency of array of extracted bids, process may follow in a two different ways: if an array of extracted bids is empty, all contracting process will go to termination as 'unsuccessful'



If an array of extracted bids is not empty, process will go to next check



Process sequence diagram (https://goo.gl/7gyszq)

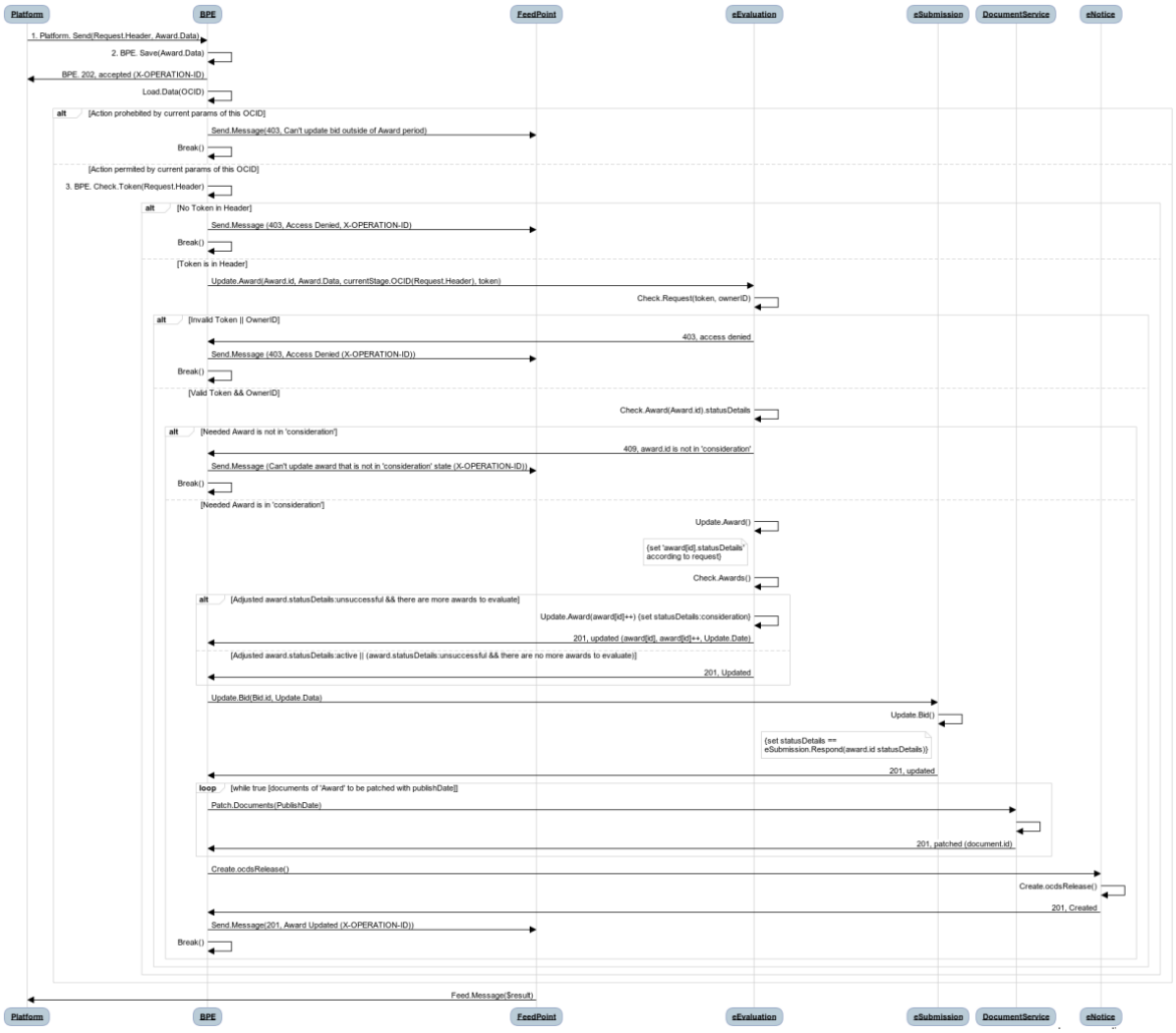
BPE-EVALUATION-AWARDING:STEP: execution of awarding stage (evaluation phase)

Update auto-generated 'Award'

Incomes

payload and params	`AWARD update model according to API documentation`
endpoint	`/do/award`

Execution diagram



Process sequence diagram (https://goo.gl/mnuVgu)

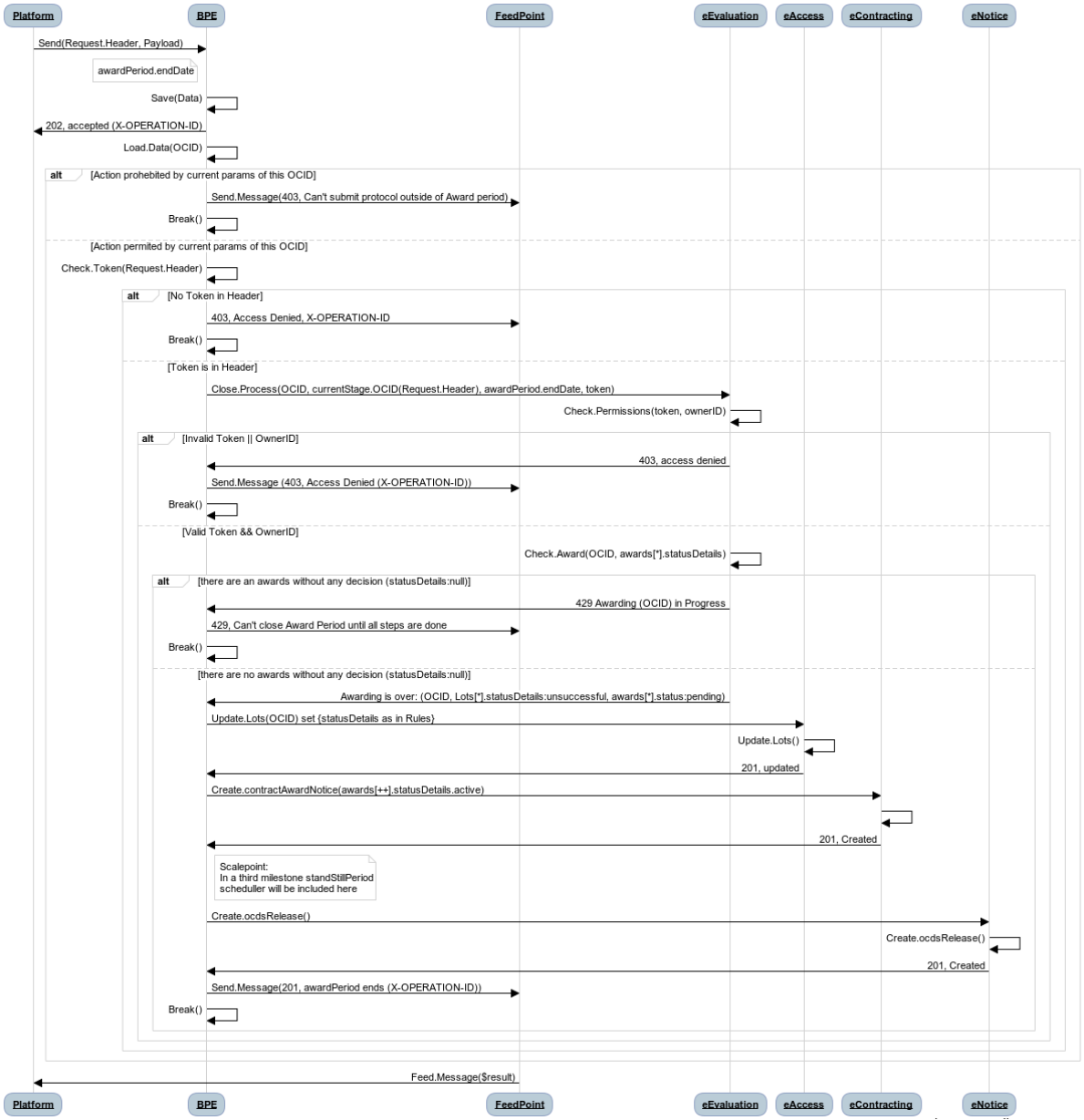
BPE-EVALUATION-AWARDING:PROTOCOL: execution of end of awarding process

Update auto-generated 'Award'

Incomes

payload and params	`PROTOCOL patch according to API documentation`
endpoint	`/do/protocol`

Execution diagram



Process sequence diagram (https://goo.gl/XZqu4H)

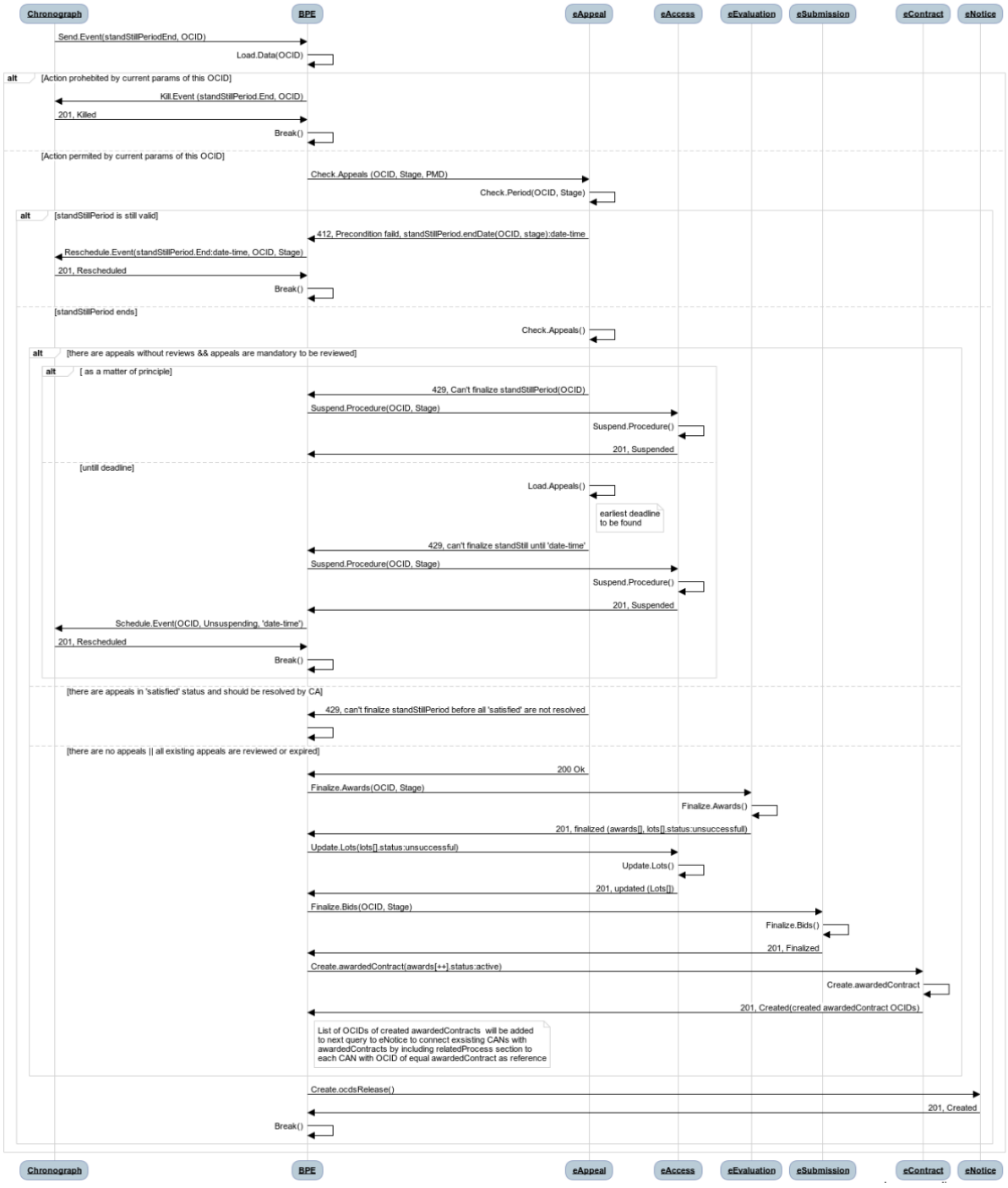
BPE-EVALUATION-AWARDING:END: Management of end of awarding stage and all phase

Close of awarding period in current phase (automated)

Incomes

payload and params	`automated internal process`

Execution diagram



Process sequence diagram (https://goo.gl/aQcDsZ)

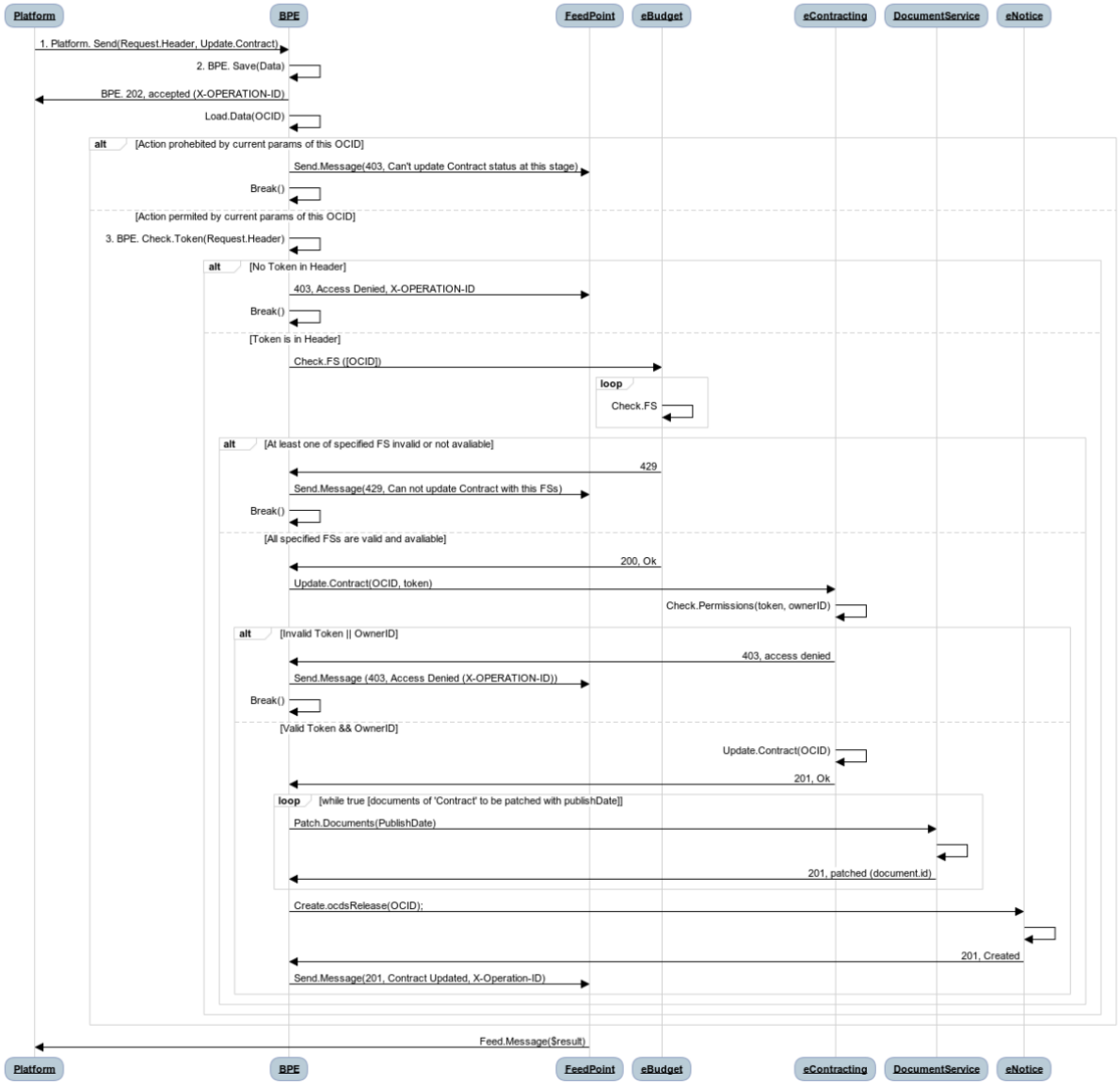
BPE-CONTRACTING-CONTRACT:PREPARATION: preparation of contract

Close of awarding period in current phase (automated)

Incomes

payload and params	`AC update model according to API documentation`
endpoint	`/do/contract`

Execution diagram



Process sequence diagram (https://goo.gl/UkMH8W)

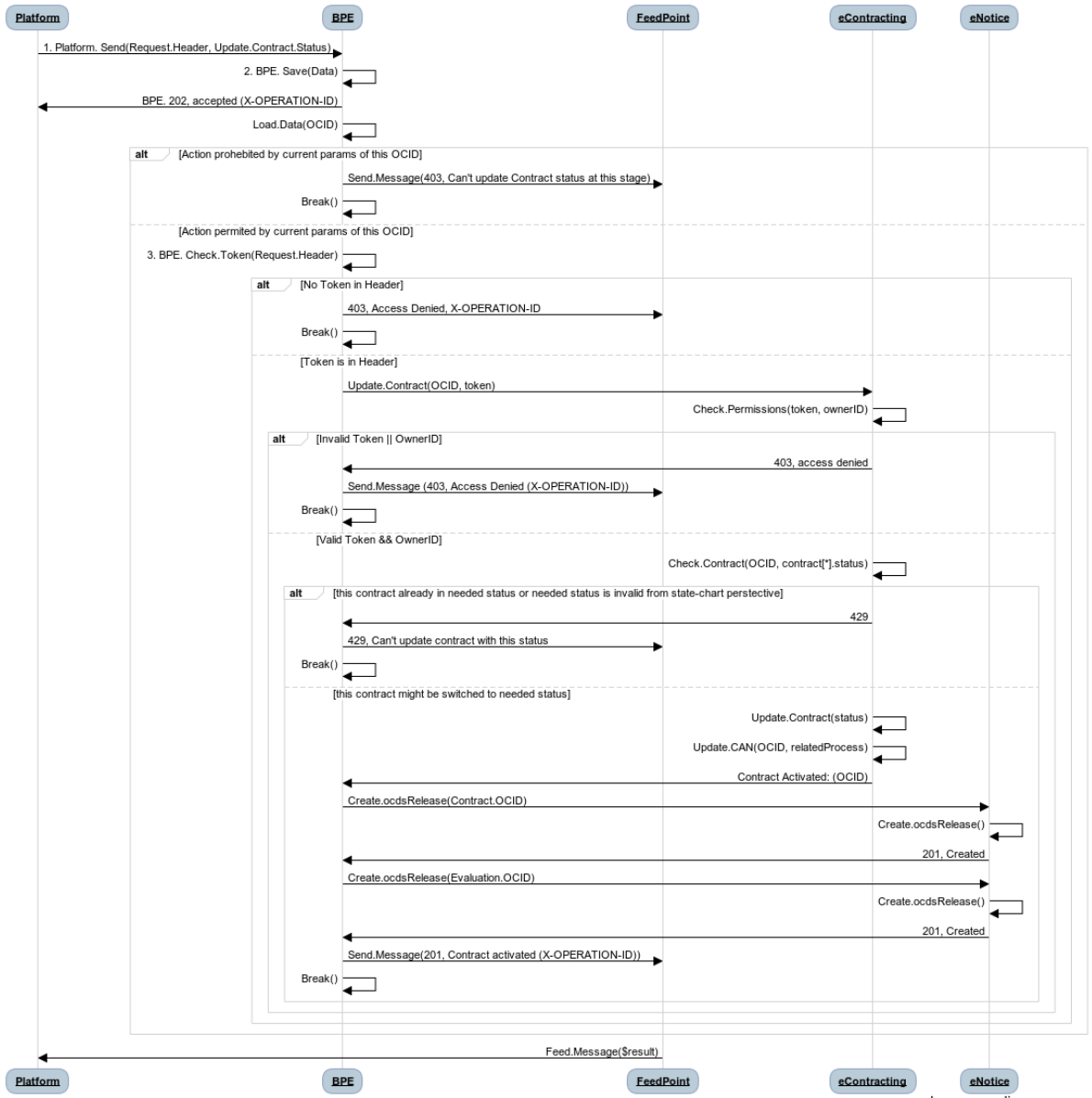
BPE-CONTRACTING-CONTRACT:ACTIVATION: contract activation and closing of evaluation phase

Close of awarding period in current phase (automated)

Incomes

payload and params	`AC patch according to API documentation`
endpoint	`/issue/contract`

Execution diagram



Process sequence diagram (https://goo.gl/UYbKwG)

7 Public API

The public API is the only interface to use the Central Business Process Engine that is the core unit of infrastructure. The API is a CQRS interface that provides programmatic access to BPE. It provides predictable URLs for accessing resources, and uses built-in HTTP features to receive commands and return responses. This makes it easy to communicate with. The API accepts JSON or form-encoded content in requests. It returns JSON content in all of its responses, including errors.

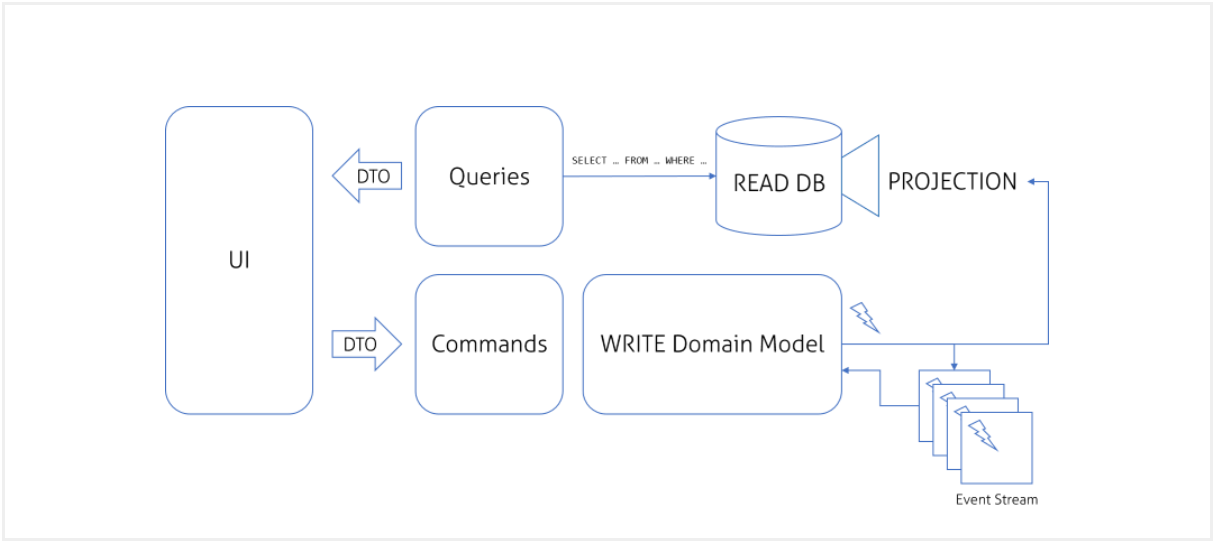
7.1 CQRS approach

In traditional architectures, the same data model is used to query and update a database. That's simple and works well for basic CRUD operations. In more complex applications, however, this approach can become unwieldy. For example, on the read side, the application may perform many different queries, returning data transfer objects (Data Transfer Objects or DTOs) with different shapes. Object mapping can become complicated. On the write side, the model may implement complex validation and business logic. As a result, you can end up with an overly complex model that does too much. Another potential problem is that read and write workloads are often asymmetrical, with very different performance and scale requirements.

Command and Query Responsibility Segregation (CQRS) - an architecture which addresses these problems by separating reads and writes into separate models, using commands to update data, and queries to read data.

- Commands should be task based, rather than data centric. Commands may be placed on a queue for asynchronous processing, rather than being processed synchronously.
- Queries never modify the database. A query returns a DTO that does not encapsulate any knowledge.

Compared to the single data model used in CRUD-based systems, the use of separate query and update models for the data in CQRS-based systems simplifies design and implementation.



For greater isolation, the read and the write data could be physically separated. In that case, the read database can use its own data schema that is optimized for queries. For example, it can store a materialized view of the data, in order to avoid complex joins or mappings. It might even use a different type of data store. If separate read and write databases are used, they must be kept in sync. Typically this is accomplished by having the write model publish an event whenever it updates the DB.

7.2 Command API

This section explain how the interfaces to the provided services are called.

```
https://bpe.HOST
```

7.2.1 Request example

All requests for Form Service are use POST-method (same as for parent Operation Service)

```
POST                          endPoint?specificParams[...]                    HTTP/1.1
Authorization:                                Bearer                              ...
X-OPERATION-ID:                                                                   ...
Content-Type:                                                            application/json
Host: bpe.HOST
```

Specific Params

Set of specific parameters required in particular case. Described below.

7.2.2 Response examples

For all valid queries, the response will have a common structure:

```
202                                                                    Accepted
Content-Type: application/json; charset=UTF-8
```

For all invalid queries, the response will have a common structure:

```
4XX
Content-Type:                    application/json;                    charset=UTF-8
{
  "errors":[
   {
     "code":                                                                  "",
     "description":                                                           ""
   }
  ]
}
```

7.2.3 Activities

This section describes the array of activities that can be performed through the API in each of the processes stages.

7.2.3.1 Budgeting

Budgeting is a stage where CA should assess and define the scope of his needs as well as disclose and describe existing fundings for each defined need. With this description CA will allow system to validate all future contracting processes from assessed needs and available funding perspective as well as control and disclose the progress of execution of this or that expenditure item, analyzing planned and announced procedures, conducted and executed or/and terminated contracts and made transactions.

All needs could be divided into separate expenditure items - groups of needs united under common classification and procuring period for each specific buyer. For example: buyer needs to procure 25 different kinds of furniture (on this stage - doesn't matter which exactly) during next year - all these needs are expenditure item «Furniture» under CPV-code 391. Thus, from technical point of view such groups, expenditure items, are aggregated data-sets, based on common CPV-code and budget period (most often - year). Each defined expenditure item, on a annual financial plans' level, has available fundings. Thus, next step of budgeting is description of such available fundings. The 'amount' available under this expenditure item for this period is the sum of all sources of funding and its amounts, identified for this expenditure item.

Expenditure Item (EI)

Create ei

```
POST /do/ei?country=...
```

country	Identifier of country of jurisdiction according to 'country' codelist

Update ei

```
POST /do/ei/ocid
```

ocid	OCID of EI to be updated

Funding Source (FS)

Create fs

```
POST /do/fs/ocid
```

ocid	OCID of parent EI

Update fs

```
POST /do/fs/ocid/fs-ocid
```

ocid	OCID of parent EI
fs-ocid	OCID of FS to be updated

7.2.3.2 Planning

Once the EI and its FSs defined and described, CA is able to plan and schedule needed procurements - contracting processes for this subject classification - CPV group. Particular EI should be indicated as a ground (meta-parent) of future contracting process. Then, depending of CAs procurement strategy, Periodic Notice or Prior Information Notice could be prepared and published. Within this notice (PN or PIN) allocated by CA himself budget must be specified as a set of identified from the list of available under used EI funding sources together with allocated amounts - separate for each selected FS. The sum of all specified amounts of all indicated funding sources is a budget of future contract that will be conducted under this contracting process.

Planning Notice (PN)

Create pn

```
POST /do/pn?country=...&pmd=...
```

country	Identifier of country of jurisdiction according to 'country' codelist
pmd	Identifier *procurementMethodDetails* according to 'pmd' codelist

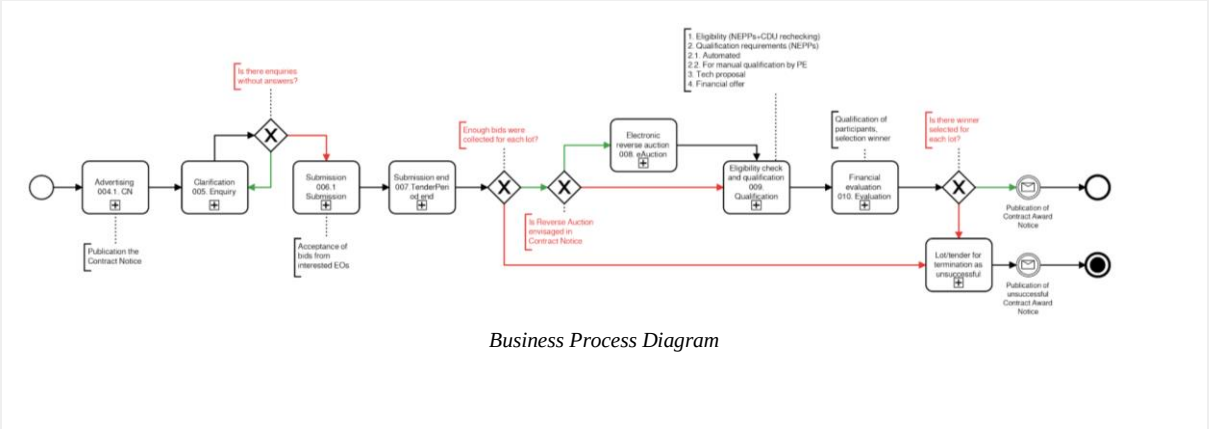
Update pn

```
POST /do/pn/ocid/pn-ocid
```

ocid	OCID of parent Contracting Process
pn-ocid	OCID of PN to be updated

7.2.3.3Announcement

Depending on type of procedure, method of procurement, geography and the legal basis, the attribute composition of the model can be adjusted, but its general logic remains unchanged for all types of procedures. The contracting process based on competitive procedure is commonly carried out in the following sequence:



Business Process Diagram

Creating a tender, CA determines key fields of the yet-to-be announced procurement, uploads tender documentation, with the requirements to the procurement object indicated as that which determines winner evaluation criteria.

Contract Notice (CN)

Create cn

```
POST /do/cn?country=...&pmd=...
```

country	Identifier of country of jurisdiction according to 'country' codelist
pmd	Identifier procurementMethodDetails according to 'pmd' codelist

Create cn based on pn

```
POST /do/pn/ocid/pn-ocid
```

ocid	OCID of parent Contracting Process
pn-ocid	OCID of PN to be updated

Update cn

```
POST /do/cn/ocid/cn-ocid
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated

7.2.3.4Clarification

Clarification period is separately distinguished in the procurement procedure during which participants can ask questions regarding the procurement requirements, demand issue resolution, and submit a claims, while CA can provide answers to questions and introduce changes into the procurement conditions. The duration of the clarification period and offer submission is determined by CA but in any case can not be less than those, prescribed by Law.

Create enquiry

```
POST /do/enquiry/ocid/cn-ocid
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated

Create answer

```
POST /do/enquiry/ocid/cn-ocid/enquiry-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
enquiry-id	ID of parent enquiry

7.2.3.5Submission

Once the clarification period is over the CA can no longer introduce changes into the Contract Notice. Economic Operators submit offers that are confidential.

Create bid

```
POST /do/bid/ocid/cn-ocid
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated

Update bid

An Economic Operator may modify its electronic bid by submitting online within submission deadlines a new information regarding previously submitted offer in accordance to the electronic submission procedures.

```
POST /do/bid/ocid/cn-ocid/bid-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
bid-id	ID of the bid

Update winning bid

```
POST /do/bidDocs/ocid/cn-ocid/bid-id
```

ocid	OCID of parent Contracting Process

cn-ocid	OCID of CN to be updated
bid-id	ID of the bid

Withdraw bid

An Economic Operator may withdraw his electronic bid by submitting online within submission deadlines a request for cancellation of previously submitted offer.

```
POST /cancel/bid/ocid/cn-ocid/bid-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
bid-id	ID of the bid

7.2.3.6Awarding

Once the submission period is over and participants can not longer submit or update their bids, system will disclose all submitted bids, generate set of qualification envelopes (awards) and launch the awarding period for this contracting process. System will automatically verify eligibility (based on available via external bus data) of each tenderer whose bid was disclosed according to rules of dispatch under current procurement method.

Those bids that have been passed eligibility check, will go to the technical qualification. All the other that failed go to automatic exclusion. In case if after submission period end there is nothing to disclose (no bids were submitted) or all submitted bids failed, system will automatically change this specific lot of this contracting process to status “unsuccessful”.

Create award

```
POST /do/award/ocid/cn-ocid
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated

Update award

CA sequentially reviews bids that are eligible considering its technical compliance, beginning with first, suggested by the system. Depending on award criteria (priceOnly/costOnly/MEAT) system will rank eligible bids and suggest them in order of admissibility: from the most to the

least acceptable by applicable criteria. If the most acceptable bid is in compliance with the CA's technical requirements, CA determines this offer as a qualified and goes to next step - financial evaluation. If it is not, CA confirms his decision to disqualify the participant, and declines such an offer. Then the system suggests to qualify the next offer from the acceptance perspective.

If qualified offer is in compliance with the CA's financial requirements, CA determines this offer as a winner. If it is not in compliance, CA confirms his decision to disqualify the offer on this (evaluation) stage, and declines such an offer. Then the systems suggests to qualify the next offer from the acceptance perspective.

If all the offers were declined by CA either on qualification step or on evaluation one and upon the completion of complaining process (review period) this specific lot of this contracting process automatically changes to 'unsuccessful'.

```
POST /do/award/ocid/cn-ocid/award-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
award-id	ID of award to be updated

Submit awarding protocol

Once winner is selected for each lot, CA submits the evaluation protocol which, in its turn, indicates the end of awarding process. As a result of submission of evaluation protocol, system will automatically generate a set of Submission for each awarded lot, change evaluation stage to 'awarded' and launch the complaining process - review period.

```
POST /do/protocol/ocid/cn-ocid/lot-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
lot-id	ID of the parent lot

Update Contract Award Notice

```
POST /do/document/ocid/cn-ocid/can-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
can-id	ID of the CAN to be updated

Confirm Contract Award Notice

```
POST /do/confirmation/can/ocid/cn-ocid/can-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
can-id	ID of the CAN to be updated

Cancel Contract Award Notice

```
POST /cancel/can/ocid/cn-ocid/can-id
```

ocid	OCID of parent Contracting Process
cn-ocid	OCID of CN to be updated
can-id	ID of the CAN to be updated

7.2.3.7 Contracting

Create awarded contract

```
POST /do/contract/ocid/ac-ocid
```

ocid	OCID of parent Contracting Process
ac-ocid	OCID of the *contract* to be updated

Update awarded contract

```
POST /do/contract/ocid/ac-ocid
```

ocid	OCID of parent Contracting Process
ac-ocid	OCID of the *contract* to be updated

Issue contract to be approved

```
POST /issue/contract/ocid/ac-ocid
```

ocid	OCID of parent Contracting Process
ac-ocid	OCID of the *contract* to be issued

Approve contract

```
POST /do/confirmation/ocid/ac-ocid/request-id
```

ocid	OCID of parent Contracting Process
ac-ocid	OCID of the *contract* to be approved
request-id	ID of initial

Activate approved contract

```
POST /activate/contract/ocid/ac-ocid
```

ocid	OCID of parent Contracting Process
ac-ocid	OCID of the *contract* to be activated

Cancel awarded contract

```
POST /cancel/contract/ocid/ac-ocid
```

ocid	OCID of parent Contracting Process
ac-ocid	OCID of the *contract* to be cancelled

7.2.4 Query API

Public Point is a service that displays records and data-sets in read-only mode. Only information allowed for publication by the current legislation is displayed (i.e. it does not show bids of the bidders until the opening session (disclosure) of the bids.

7.2.4.1 AccessPoint

All of the document uploading API endpoints follow the same set of rules and uses single access point.

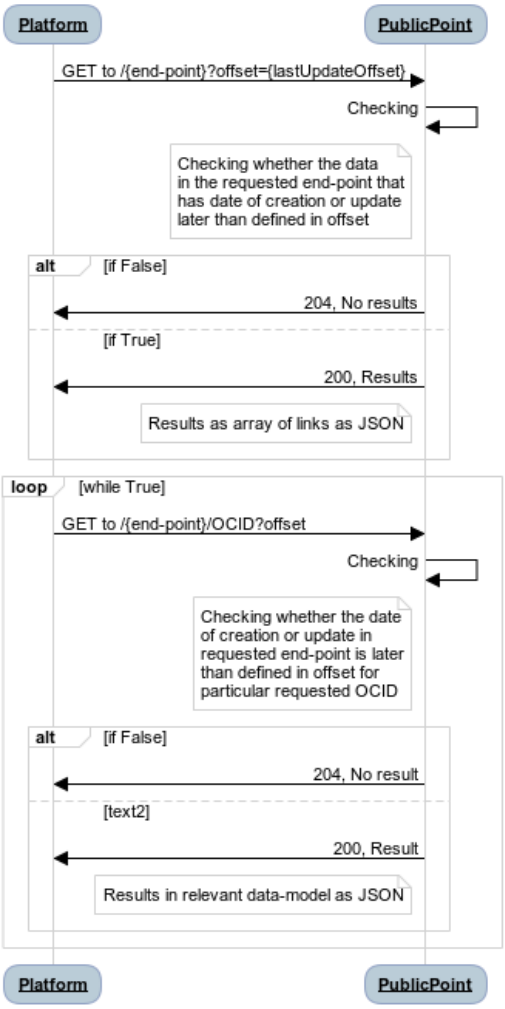
```
http://public.HOST
```

7.2.4.2 Flow Description

File to be registered

DS performs following actions:

1. The file size is checked - if the value is exceeded, the process is thrown
2. The file format is checked for a valid format
3. In case of a valid format, the file is saved:
   a. description is stored in the database according to the model
   b. Returns file descriptions to the client according to the attribute composition



Process Sequence Diagram

Retrieving data

To retrieve a list of data or individual entity data-set use GET-request parameterized with one or more options sent to corresponding end-point.

- offset - timestamp of latest updates
- limit - quantity of elements per each package of data
- order - criteria for sorting results

Getting list of all data

By default all data returned from public point are sorted by modification time (from oldest to newest).

Limiting number of Tenders returned

You can control the number of data entries in the entity feed (batch size) with limit parameter. If not specified, data is being returned in batches of 100 elements.

```
GET                                /get/tenders?offset=...&limit=20&order=asc&mode=history

200                                                                                OK
Content-Type: application/json; charset=UTF-8

{
  "data":                                                                           [
    {
      "ocid":                                                                     "",
      "date":                                                                      ""
    }
  ],
  "offset":""
}
```

Batching

The response contains ‘offset’ element. This is the parameter you have to add to the original request you made to get next page. If next page request returns no data (i.e. empty array) then there is little sense in fetching further pages.

Synchronizing

It is often necessary to be able to synchronize changes of Central Unit with local database. The default sorting “by modification date” together with Batching mechanism allows one to implement synchronization effectively. The synchronization process can go page by page until there is no new data returned. Then the synchronizer has to pause for a while to let Central Unit register some changes and attempt fetching subsequent page.

Getting individual entity information

To get a individual entity data set (as Record Package) provide your GET-request with unique ID of particular entity:

```
GET /get/tenders/ocds-000-00001?offset=...
```

8 Folder structure

Kotlin development:

module_name

|- docs

|- src

|- main

|- kotlin/com/procurement/*module_name*

|- application

|- config

|- controller

|- dao

|- databinding

|- domain/model

|- enums

|- exception

|- infrastructure

|- lib

|- model

|- service

|- utils

|- *module_aplication*.kt

|- resources

|- script

|- dockerfile

|- test

|- pom.xml