DG Digitale Transformatie
FOD Beleid en Ondersteuning
DG Transformation digitale
SPF Stratégie et Appui

BeStAddress User Guide
Part 3: MFT services

Date: 06/12/2022
Version: 2.5

Version History

Version	Date	Modified by	Modification
1.0	09/05/19	Gert De Jonge	First version
2.0	23/05/2022	Luc Mertens	Adaptation to new services
2.1	02/08/2022	Luc Mertens	Adding output to partners in MFT services
2.2	21/10/2022	Eddy Corthouts	Reviewed and updated section 4.2, “Address mutations file service”
2.3	28/10/2022	Eddy Corthouts	Split original BeSt user guide into 3 separate documents
2.4	10/11/2022	Eddy Corthouts	Minor changes
2.5	6/12/2022	Eddy Corthouts	Section 2.1: update availability times of download files

Instructions for maintaining the document:

In case a new XSD is implemented change the reference in par. 2.2 , “S3–9 - FullDownloadService”.

Conventions

Font	use
Italic	accentuation

Contact information

Service Owner	Sebastiaan Taes Sebastiaan.Taes@bosa.fgov.be
Service Desk	ServiceDesk@bosa.fgov.be +32 78 150312 +32 2 2129674
Service Release Date	TBD

All BeSt services are subject to the latest BOSA FSB Terms & Conditions, such as described in this document. The document describes the governance principles of the BOSA Service Bus as well.

Table of contents

List of Figures

Glossary

These are terms specific to this document, general terms known inside the BeSt environment are not added.

Term	Description
Object	Object is a general term, it represents an independent element such as building, parcel, Address, Municipality, StreetName, PostalInfo…
Class	Template or blueprint that is used in modelling techniques to describes an object.
(BeSt) Identifier	Combination of the namespace, objectIdentifier and versionIdentifier which uniquely identify an object.
Entity	Representation of the BeSt object in the real world. The entity is identified by a complete BeStIdentifier. (So every version of an object is an entity) The entity corresponds with 1 record in our dataset.
Component	A sub part of an address or a StreetName. (Objects that are linked to another object) Address has following sub parts: Municipality, StreetName, PostalInfo, PartOfMunicipality A StreetName has 1 sub part: Municipality Municipality, PostalInfo and PartOfMunicipality don't have sub parts.
Linkable & Linked entity	Entities that have no components but are linked together because there is an address that defines this link. It concerns Municipality, PostalInfo and PartOfMunicipality. Linked entities are always from another type considering 1 address can contain only 1 entity of each type.
History Chain	The history chain allows to retrieve the history of a particular Address (or address component), it consists of a chain of entities that make up the history of the address (or address compontent)
Parameter	(Input) item of the request interface.
Sub-parameter	Parameter that is part of a “combined” parameter.
Enumerated parameter	Parameter with a limited number of allowed values. This includes all Boolean parameters.
(output) Field	Output field in the reply interface of a service.
Prefix mun	For referrals to municipality in the name of a property, it is prefixed with mun
Prefix pom	For referrals to PartOfMunicipality in the name of a property, it is prefixed with pom
Prefix post	For referrals to PostalInfo in the name of a property, it is prefixed it with post
Prefix street	For referrals to StreetName in the name of a property, it is prefixed with street
predecessor	The BeSt-identifier of the record that will be replaced. This will only be filled in on the ‘Add’ element
successor	The BeSt-identifier of the record that is the replacemen,t of the current record. This will only be filled in on the ‘Update’ element
MFT	Managed File Transfer
eventType	Type of change . This can be filled on or left blank. Each Region can have their own list of events. The Lists are added in the extensions.
Correction	A cosmetic change or a spelling correction

1 Introduction

BeSt stands for “Belgian Streets”. The BeSt services provide address information on a federal level based on the three regional address master data sets from Brussels, Flanders and Wallonia.

BeSt was developed based on the agreement from 17 juli 2019 between the Federal government, the Brussels region, the Flanders region and the Wallone region on how to reference and link address data.
In addition, the following organizations have participated in the development and implementation of BeSt:

- The National Geographic Institute (NGI)
- The General Administration of the Patrimony Documentation (AAPD) from the FPS Finance
- The National Registry (NR) from the FPS Internal Affairs
- Statistics Belgium from the FPS Economy
- The Crossroad Bank for Enterprises (CBE) from the FPS Economy
- The Directorate general Security and Prevention from the FPS Internal Affairs
- The FPS Governance and Support (BOSA)
- The Agency for Administrative Rationalization (DAV)
- The supplier of the universal postal services

1.1 Intended Audience

This document is intended for any analyst or developer who wants to make use of the BoSa BeSt Address services.

1.2 Available documentation

The next table provides an overview of the documentation available:

User guide	**Purpose**
1 BeSt_Userguide_INTRO_and_DATA	Provides an overview of the BeSt application and describes the BeSt data, including the data model and the different data entities with their elements.
2 BeSt_Userguide_WEB_services	Describes the webservices that are available to the consumer to consult the BeSt address data
3 BeSt_Userguide_MFT_services (this document)	Describes the Managed File Services that are available to the consumer to obtain a full download file of BeSt addresses or to obtain daily mutations

Purpose of this document

This document describes the *BeSt Managed File Transfer Services (MFT services)*.

The MFT services allow the user to download address information by means of files. The MFT services are aimed at customers that want to download a lot or all address information or that want to maintain a local copy of the addresses in their own database.

For an overview of the BeSt application, please refer to the document “1. BeSt_Userguide_INTRO_and_DATA”.

2 MFT Services

The following MFT services are available.

MFT services	**Description**	**Availability Date**
S349 – FullDownloadService	Download of all address information	15/05/2019
S350 – AddressMutationsFileService	Download of daily mutations	*Q2 2022*
SXXX – StructuralAnomalyFileService	Download a weekly file with structural anomalies	*Q3 2022*

The MFT services fetch their information from the BOSA address database that is a copy of the authentic sources of the regions. For an overview of the complete process, please refer to “1. BeSt_Userguide_INTRO_and_DATA”

Below, these services are described in further detail.

2.1 General information

Availability to the Consumer

Download for registered users

The files will be available to registered users on the BOSA server for downloading at the following time:

Service	File Name	Frequency	Time
S349 – FullDownloadService	FDBelgiumxxxxxxxx.zip	Weekly	Sunday 03:00 AM
S3–0 - AdresMutationsFileService	MBelgiumxxxxxxxx.zip	Daily on weekdays	Tue, We, Th, Fr, Sa – 03:00 AM
SXXX – StructuralAnomalyFileService	XXXX	Weekly	Mo 06:00 AM

Important: it is necessary that every client that uses the Full Download uses the mutations . All the changes that were made during the week (between two full download dates) will be available in the mutations these are necessary to keep one's own address database up-to-date.

Download from Open Data website

The full download will be available on the Open Data website at the following time:

Service	File Name	Frequency	Time
S349 – FullDownloadService	FDBelgiumxxxxxxxx.zip	Weekly	Sunday 04:00 AM

The AddressMutationsFileService and the StructuralAnomalyFileService are not available on the OpenData platform.

Retention Policy

BOSA keeps the BeSt download files available for 30 days.

Processing of incoming Region files by BOSA

Processing of Region files by BOSA
BOSA will pick up 1 full download per week on Sundays from 23:00 PM from each of the regions.
BOSA will pick up mutations on a daily basis (week days) from 23:00 onwards from each of the regions.
BOSA does not delete the files from the region pick-up sites.

BOSA will prepare the consumer download files by 01:00

BOSA processing of Region files encompasses:

- validate the files are conform with the XSD's
- zip the files into a single zip file for downloading by consumers

- This process takes about 15 min.

As the files are processed in the night, it is assumed that no mutations can occur during the update window.

2.2 S3–9 - FullDownloadService

Main functionality

The Full Download allows a user to download by means of MFT a zip file that contains one file for each entity of the BeSt data model per region. These files contain for each Region the latest versions of the following entities:

- municipalities
- streetnames
- addresses
- partofmunicipalities (Wallonia only)
- postalinfo

Each region produces 4 or 5 separate files containing the latest version of that specific entity. If on a certain day, no files would be received from a region for one or more entities, BOSA will place the previous version received in the BeSt full download for those entities.

Data Structure of files

The files are structured and based on the BeSt address model (see section 2) but in a slightly different manner:

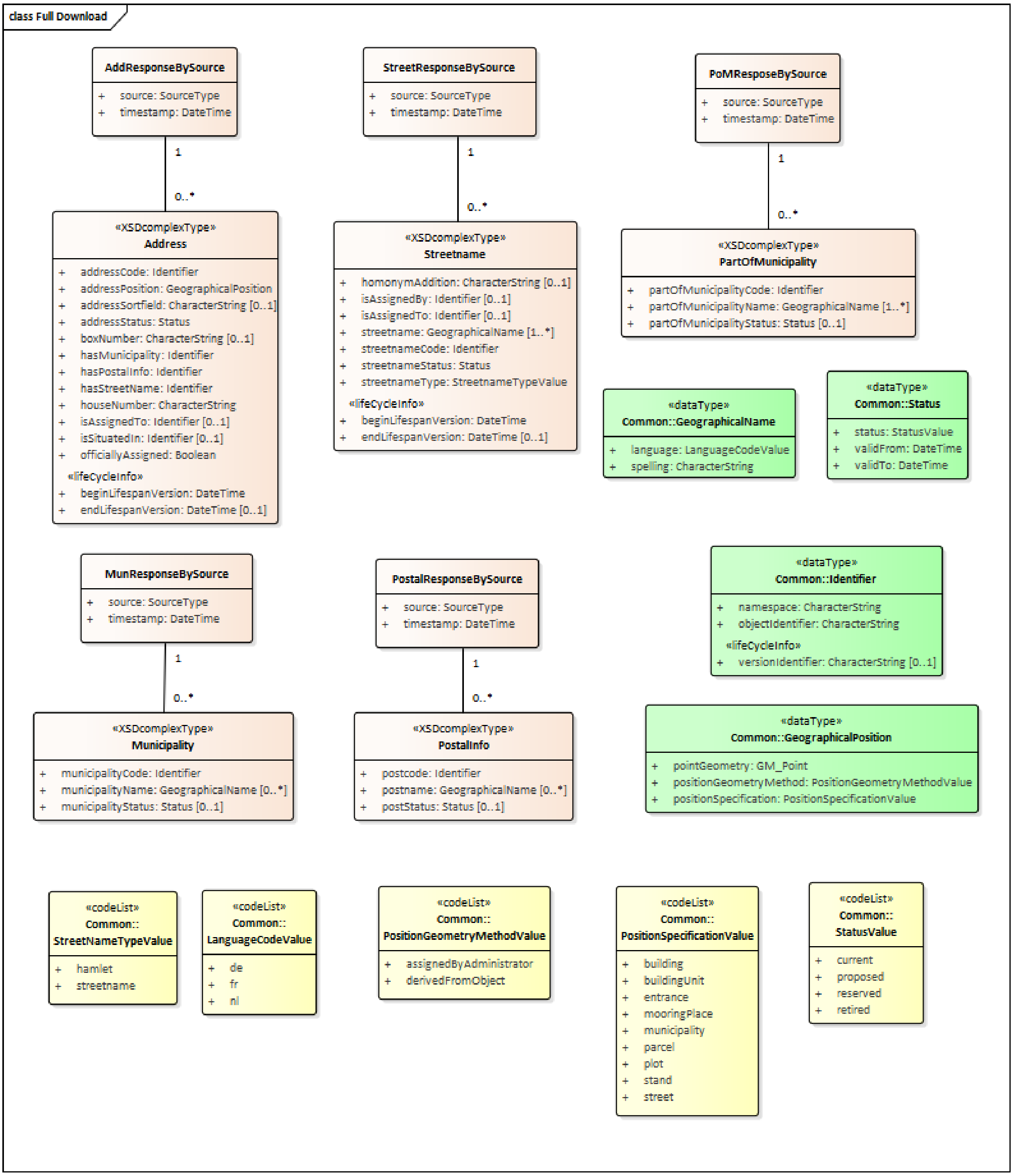

class Full Download
AddResponseBySource
+ source: SourceType
+ timestamp: DateTime
1
0..*
«XSDcomplexType»
Address
+ addressCode: Identifier
+ addressPosition: GeographicalPosition
+ addressSortfield: CharacterString [0..1]
+ addressStatus: Status
+ boxNumber: CharacterString [0..1]
+ hasMunicipality: Identifier
+ hasPostalInfo: Identifier
+ hasStreetName: Identifier
+ houseNumber: CharacterString
+ isAssignedTo: Identifier [0..1]
+ isSituatedIn: Identifier [0..1]
+ officiallyAssigned: Boolean
«lifeCycleInfo»
+ beginLifespanVersion: DateTime
+ endLifespanVersion: DateTime [0..1]
StreetResponseBySource
+ source: SourceType
+ timestamp: DateTime
1
0..*
«XSDcomplexType»
Streetname
+ homonymAddition: CharacterString [0..1]
+ isAssignedBy: Identifier [0..1]
+ isAssignedTo: Identifier [0..1]
+ streetname: GeographicalName [1..*]
+ streetnameCode: Identifier
+ streetnameStatus: Status
+ streetnameType: StreetnameTypeValue
«lifeCycleInfo»
+ beginLifespanVersion: DateTime
+ endLifespanVersion: DateTime [0..1]
PoMResposeBySource
+ source: SourceType
+ timestamp: DateTime
1
0..*
«XSDcomplexType»
PartOfMunicipality
+ partOfMunicipalityCode: Identifier
+ partOfMunicipalityName: GeographicalName [1..*]
+ partOfMunicipalityStatus: Status [0..1]
«dataType»
Common::GeographicalName
+ language: LanguageCodeValue
+ spelling: CharacterString
«dataType»
Common::Status
+ status: StatusValue
+ validFrom: DateTime
+ validTo: DateTime
MunResponseBySource
+ source: SourceType
+ timestamp: DateTime
1
0..*
«XSDcomplexType»
Municipality
+ municipalityCode: Identifier
+ municipalityName: GeographicalName [0..*]
+ municipalityStatus: Status [0..1]
PostalResponseBySource
+ source: SourceType
+ timestamp: DateTime
1
0..*
«XSDcomplexType»
PostalInfo
+ postcode: Identifier
+ postname: GeographicalName [0..*]
+ postStatus: Status [0..1]
«dataType»
Common::Identifier
+ namespace: CharacterString
+ objectIdentifier: CharacterString
«lifeCycleInfo»
+ versionIdentifier: CharacterString [0..1]
«dataType»
Common::GeographicalPosition
+ pointGeometry: GM_Point
+ positionGeometryMethod: PositionGeometryMethodValue
+ positionSpecification: PositionSpecificationValue
«codeList»
Common::
StreetNameTypeValue
+ hamlet
+ streetname
«codeList»
Common::
LanguageCodeValue
+ de
+ fr
+ nl
«codeList»
Common::
PositionGeometryMethodValue
+ assignedByAdministrator
+ derivedFromObject
«codeList»
Common::
PositionSpecificationValue
+ building
+ buildingUnit
+ entrance
+ mooringPlace
+ municipality
+ parcel
+ plot
+ stand
+ street
«codeList»
Common::
StatusValue
+ current
+ proposed
+ reserved
+ retired

Figure 1, 'Full download data structure'

Currently, the XSD file used has version 23.3.1

File Naming Conventions

Region files

Each region produces 4 separate files:

The name of the files follows the following convention: RegionName+ entity + date + coordinate-system (e.g. L72)

<u>Example for Brussels</u>:

- BrusselsAddress20190319L72.zip
- BrusselsMunicipality20190319L72.zip
- BrusselsPostalinfo20190319L72.zip
- BrusselsStreetname20190319L72.zip

There is one additional file that only Wallonia provides, but not the other regions:

WalloniaPartOfMunicipalityxxxxxxxx.zip

BOSA files

BOSA combines the zipfiles from the 3 regions into a single zipfile.

The naming convention for BOSA's zipfile is

"BeStAddress_" + UseCaseName (abbreviation) + 'Belgium' + Date

Full Download:_ example: BeStAddress_**FD**Belgium20200319.zip

2.3 S3–0 - AddressMutationsFileService

Mutations are “Add” and “Update” transactions to an individual record of any kind (addresses, streetNames, Municipalities, PostalInfo, partOfMunicipality). Each region creates a mutations file on a daily basis (weekdays) which contains all mutations of that region for that day.

The mutations files are made available on a daily basis and users can use them as a complement to the weekly full download to keep their own address database up-to-date on a daily basis.

Bosa combines these 3 files into 1 large file with 3 sections. In case a region has no mutations file on a certain day, the combined file will contain no data for that region that day. (Mutatins will be reported only once)

Notes:

- An individual mutations record concerns 1 version of a BeSt Object.
- partOfMunicipality is only provided by Wallonia (not used in Brussels or Flanders)
- For Flanders, BOSA creates a standard mutations file based on sync feed events made available by Flanders; The consumer will receive a standard mutations for Flanders as is the case for the other regions.
- When comparing the mutations data with the full download data, there are 2 important differences:
  - The mutations file contains also records that are no longer valid (retired, validTo filled in).
  - The data in the mutations file includes predecessor and successor identifications: they tell the history of entities.

2.3.1 Types of mutations

A mutation is the creation of a new record or the update of an existing record. The latter includes the archiving of a record by adding a “validTo” date.

Add

An ‘Add’ element transaction occurs when:

- a new object is created in the real world, e.g. new address, new address component (Streetname, Municipality, PostalInfo, PartOfMunicipality) or a new version of a Streetname
- A new object is created as a consequence of a change of one or more data elements in that same object (e.g. status change, name change, etc.)

Update

Updates occur for entities that already exist. The identifier of the entity should already be present in your database, it is not created today.

Examples:

- Changes to Address
  - Change of the BeStIdentifier of a component
    - Municipality
    - PostalInfo

        - Streetname
        - PartOfMunicipality
    - Change of the value of an field
        - houseNumber
        - boxNumber
        - addressPosition (coordinates)
    - Change of status
    - Replacement of the entire object
- Changes to Address Components (Streetname, Municipality, PartOfMunicipality, PostalInfo)
    - Change of field value
    - Change of status
    - Replacement, Split or Merge of objects
- Updates on
    - Predecessor
    - Successor

-

Examples

Example 1: A new street is created

Events: first, the name is set and reserved and later accepted by the local government.

EVENT DATE	EVENT	RESULT
1/11/2009	New street S0 = Reserved as street	S0 status = reserved
1/01/2010	S0 is accepted by the local government	S1 status = current

The result in the mutation file on 2/11/2009 will contain following data.

ACTION	BEST IDENTIFIERS	FIELDS	STATUS	VALID FROM	VALID TO	PREDECESSOR	SUCCESSOR	EVENT	EVENT DATE
ADD	S0	All fields filled in	Reserved	1/11/2009					1/11/2009

The result in the mutation file on 02/01/2010.

ACTION	BEST IDENTIFIERS	FIELDS	STATUS	VALID FROM	VALID TO	PREDECESSOR	SUCCESSOR	EVENT	EVENT DATE
ADD	S1	Copy all fields of S0 except for status	Current	1/01/2010		S0			1/01/2010
UPD	S0	All field fields filled in	Reserved	1/11/2009	1/01/2010		S1		1/01/2010

Example 2: Addresses are replaced by other Addresses with a new houseNumber

This will result in a new BeSt-Identifier

EVENT DATE	**EVENT**
1/01/2014	A35 replaced by A45
	A36 replaced by A46
	A37 replaced by A47

The result in the mutation file will contain following data

ACTION	**BEST IDENTIFIERS**	**FIELDS**	**STATUS**	**VALID FROM**	**VALID TO**	**PREDECESSOR**	**SUCCESSOR**	**EVENT**	**EVENT DATE**
ADD	A45	All fields	Current	1/1/2014		A35			1/01/2014
ADD	A46	All fields	Current	1/1/2014		A36			1/01/2014
ADD	A47	All fields	Current	1/1/2014		A37			1/01/2014
UPD	A35	All fields	Current	1/01/2012	1/01/2014		A45		1/01/2014
UPD	A36	All fields	Current	1/01/2012	1/01/2014		A46		1/01/2014
UPD	A37	All fields	Current	1/01/2012	1/01/2014		A47		1/01/2014

Example 3: a street gets renamed

EVENT DATE	**EVENT**
	In Street S1 we have addresses A10, A11 and A12
1/01/2011	S1 is renamed into S2
	A10, A11 and A12 are respectively replaced by A20, A21 and A22

The result in the mutation file will contain following data as an ‘Update’ element

ACTION	BEST IDENTIFIERS	FIELDS	STATUS	VALID FROM	VALID TO	PREDECESSOR	SUCCESSOR	EVENT	EVENT DATE
ADD	S2	Copy all fields of S1	Current	1/01/2011		S1			1/01/2011
ADD	A20	Copy all fields of A10 except Streetname ID's	Current	1/01/2011		A10			1/02/2011
ADD	A21	Copy all fields of A11 except Streetname ID's	Current	1/01/2011		A11			1/02/2011
ADD	A22	Copy all fields of A12 except Streetname ID's	Current	1/01/2011		A12			1/02/2011
UPD	A10	All fields	Current	1/01/1990	1/01/2011		A20		1/02/2011
UPD	A11	All fields	Current	1/01/1990	1/01/2011		A21		1/02/2011
UPD	A12	All Fields	Current	1/01/1990	1/01/2011		A22		1/02/2011
UPD	S1	All fields	Current	1/01/1990	1/01/2011		S2		1/01/2011

2.3.2 Cascade mutations

Cascade mutations are mutations that occur in an object due to a mutation in another, linked object. The next table shows how a change in one BeSt objects can result in changes in other BeSt objects.

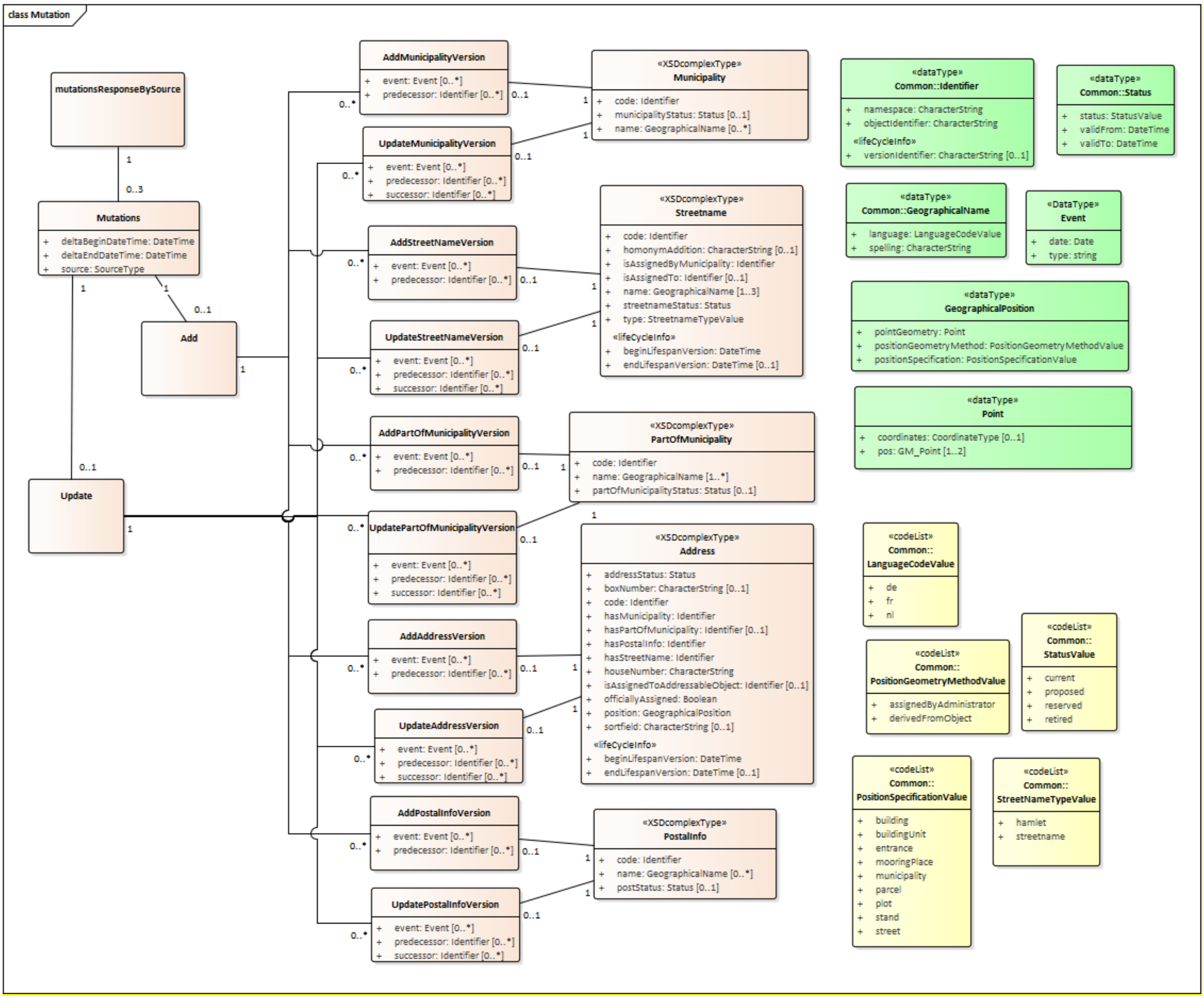
	Changed element	Impact on Address identifier (=BeSt identifier)	Impact on Address Component identifier (Streetname) (=BeSt identifier)	Impact on Address Component identifier (Municipality, PostalInfo, PartOfMunicipality) (=BeSt identifier)
Address	**addressCode (=BeSt identifier)**	Change	▨	▨
	houseNumber	Change	▨	▨
	houseNumber (correction) (*)	Change	▨	▨
	boxNumber	Change	▨	▨
	boxNumber (correction)	Change	▨	▨
	addressPosition	Change	▨	▨
	addressStatus	Change	▨	▨
Municipality	**municipalityCode (=BeSt identifier)**	Change	Change	Change
	municipalityName.spelling	▨	▨	N/A
	municipalityName.spelling (correction)	Has change in BeStidentifier or only on element municipalityCode (**)	no BeStidentifier change Only update on element isAssignedBy	Change (**)
	municipalityStatus	TBD (Future phase)(***)	TBD (Future phase)(***)	TBD (Future phase) (***)
Streetname	**streetnameCode (=BeSt identifier)**	Change	Change	▨
	streetname.spelling	Change	Change	▨
	streetname.spelling (correction)	Change	Change	▨
	streetnameStatus	Change	Change	▨
	homonymAddition	Change	Change	▨
PostalInfo	**postcode**	Change	▨	Change
	postname	▨	▨	N/A
	postStatus	TBD (Future phase) (***)	🟥	TBD (Future phase) (***)
PartOfMunicipality	**partOfMunicipalityCode (=BeSt identifier)**	Change	▨	Change
	partOfMunicipalityname.spelling	▨	▨	N/A
	partOfMunicipalityname.spelling (correction)	Change	▨	Change
	partOfMunicipalityStatus	Change	🟥	Change

(*) These are small changes applied to a string where a human being can unambiguously conclude that the same thing is targeted (spelling/typo/cosmetic corrections, but no fundamental change).Corrections on address are still under investigation with the

(**) FL updates version of Municipality but does not update the address (as in their applications, the version ID is not part of the unique key in their system) but We do so we send an update of the address with the new versionId of the Municipality but no new version of the address.

(***) Possible future input

2.3.3 Mutations data structure



class Mutation
mutationsResponseBySource
1
0..3
Mutations
+ deltaBeginDateTime: DateTime
+ deltaEndDateTime: DateTime
+ source: SourceType
1
1
0..1
Add
1
0..1
Update
1
AddMunicipalityVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
UpdateMunicipalityVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
+ successor: Identifier [0..*]
AddStreetNameVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
UpdateStreetNameVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
+ successor: Identifier [0..*]
AddPartOfMunicipalityVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
UpdatePartOfMunicipalityVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
+ successor: Identifier [0..*]
AddAddressVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
UpdateAddressVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
+ successor: Identifier [0..*]
AddPostalInfoVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
UpdatePostalInfoVersion
+ event: Event [0..*]
+ predecessor: Identifier [0..*]
+ successor: Identifier [0..*]
0..*
0..1
1
«XSDcomplexType»
Municipality
+ code: Identifier
+ municipalityStatus: Status [0..1]
+ name: GeographicalName [0..*]
«XSDcomplexType»
Streetname
+ code: Identifier
+ homonymAddition: CharacterString [0..1]
+ isAssignedByMunicipality: Identifier
+ isAssignedTo: Identifier [0..1]
+ name: GeographicalName [1..3]
+ streetnameStatus: Status
+ type: StreetnameTypeValue
«lifeCycleInfo»
+ beginLifespanVersion: DateTime
+ endLifespanVersion: DateTime [0..1]
«XSDcomplexType»
PartOfMunicipality
+ code: Identifier
+ name: GeographicalName [1..*]
+ partOfMunicipalityStatus: Status [0..1]
«XSDcomplexType»
Address
+ addressStatus: Status
+ boxNumber: CharacterString [0..1]
+ code: Identifier
+ hasMunicipality: Identifier
+ hasPartOfMunicipality: Identifier [0..1]
+ hasPostalInfo: Identifier
+ hasStreetName: Identifier
+ houseNumber: CharacterString
+ isAssignedToAddressableObject: Identifier [0..1]
+ officiallyAssigned: Boolean
+ position: GeographicalPosition
+ sortfield: CharacterString [0..1]
«lifeCycleInfo»
+ beginLifespanVersion: DateTime
+ endLifespanVersion: DateTime [0..1]
«XSDcomplexType»
PostalInfo
+ code: Identifier
+ name: GeographicalName [0..*]
+ postStatus: Status [0..1]
«dataType»
Common::Identifier
+ namespace: CharacterString
+ objectIdentifier: CharacterString
«lifeCycleInfo»
+ versionIdentifier: CharacterString [0..1]
«dataType»
Common::Status
+ status: StatusValue
+ validFrom: DateTime
+ validTo: DateTime
«dataType»
Common::GeographicalName
+ language: LanguageCodeValue
+ spelling: CharacterString
«DataType»
Event
+ date: Date
+ type: string
«dataType»
GeographicalPosition
+ pointGeometry: Point
+ positionGeometryMethod: PositionGeometryMethodValue
+ positionSpecification: PositionSpecificationValue
«dataType»
Point
+ coordinates: CoordinateType [0..1]
+ pos: GM_Point [1..2]
«codeList»
Common::
LanguageCodeValue
+ de
+ fr
+ nl
«codeList»
Common::
StatusValue
+ current
+ proposed
+ reserved
+ retired
«codeList»
Common::
PositionGeometryMethodValue
+ assignedByAdministrator
+ derivedFromObject
«codeList»
Common::
PositionSpecificationValue
+ building
+ buildingUnit
+ entrance
+ mooringPlace
+ municipality
+ parcel
+ plot
+ stand
+ street
«codeList»
Common::
StreetNameTypeValue
+ hamlet
+ streetname

Figure 2, ‘Mutations file data structure XSD v24.2’

Types of records in the XSD

Add

Data provided:

- BeSt-Identifier is completely filled
- All fields are filled
- All BeSt-Identifiers of its components are completely filled
- Status is filled in for Address and Streetname
- validFrom date is filled in
- List of all Predecessors for the object for this event (if applicable)
- EventDate

Additional data provided depend on the availability of the data from region:

- status for Municipality, PartOfMunicipality, Postalinfo
- EventType

Update

Data provided:

- The BeSt-Identifier is completely filled in
- All fields are filled in
- All BeSt-Identifiers of its components are completely filled in
- Status is filled in (for Address and Streetname)
- validFrom date is filled in
- validTo date (if applicable)
- List of all Successors for the object for this event (if applicable)
- List of Predecessors for the object for this event (if applicable)
- EventDate

Additional data provided depending on availability from region:

- status for Municipality, PartOf Municipality, Postalinfo
- EventType

Validity rules for data

These are rules that the XML adheres to. They are enforced by the regions.

- When a replacement happens of an Address or an Address Component (new version), neither a time gap nor a time overlap is allowed with regards to the validity periods of the objects involved:
  Between the validTo date of the replaced object and the validFrom date of the replacing object there should be no gap.

  E.g.: The boxNumber of Address 1 is changed on 01/01/2010
  Address A version1 (V1) is replaced by Address A version 2 (V2)
  The validTo date of Address V1 will be set to 01/01/2010 and the validFrom date of Address V2 will be set to 01/01/2010
  This means that up to 31/12/2009 23:59:59 Address V1 was the correct Address for the object and as of 01/01/2010 00:00:00 Address V2 will be the correct Address for the object.

- When there is a Split or a Merge of an Address component, no time gap is allowed between the validity periods of the objects, but a time overlap is possible:

  E.g.: Streetname1 and Streetname2 are existing since 01/01/2000 .
  The Streetname1 is Merging with Streetname2 on 01/01/2011 (Streetname1 ceases to exist)
  The validTo date of Streetname1 will be set to 01/01/2011 but the validFrom date of Streetname2 still is 01/01/2000, this will not change.

- An Object with 'Retired' status will only occur when it doesn't exist anymore in the 'real world'.

  E.g. :
  - when you have a Streetname Kapelstraat and Kerkstraat and where these streets are situated they decide to make a new area with new streets then the Kapelstraat and Kerkstraat will stop existing and get the status retired. The same with a building when the building is destroyed and 3 apartments will be created. so not a administrative change.

  - A Municipality is retired when it merges with or splits into another Municipality and is not used anymore.

- When a replacement, split or merge occurs, all involved object changes (Address and its components) will arrive in the same file (if possible):

  - When a Streetname, Postalinfo or PartOfMunicipality record is no longer valid (validTo date is filled in), all Addresses referring to this record will be updated as well.

  - When a Municipality record is no longer valid (validTo date is filled in), all Addresses and Streetnames referring to this Municipality record will be updated as well.

- When an Address Component is no longer valid (filled validTo) then all Addresses with reference to this component should be no longer valid (filled validTo).
  e.g. In case of a Streetname change, all Addresses referring to that Streetname will have an event that indicate an Address change.

- When due to asynchronous processing of data it is possible not all information is sent the same day this can result in a missing link between old and new record. In 99.99% this will be resolved the next day with updates to these records.

- An administrative correction will be reported with the eventType ‘CORRECTION’. For all other records this field will be blank in de first phase. It might be this will be filled later on.

- If the AddressableObjectId is known we will send in the field isAssignedTo of the Address element. For the time being it is only for Flanders and informational. It will only contain namespace and objectIdentifier.

2.3.4 Structure of mutations records

2.3.4.1 Add & update address

Field	Description	Type	Max length	Min. Occurs	Max. Occurs
code		Identifier address			
code.namespace	Namespace of the address	CharStringType	55	1	1
code.objectIdentifier	ObjectIdentifier of the address	CharStringType	10	1	1
code.versionIdentifier	VersionIdentifier of the address	CharStringType	20	1	1
position		GeographicalPositionType		1	1
position.pointGeometry.point	Identifier of the point	Combi field with following attributes			
position.pointGeometry.point.pos		Lambert 72 format			
position.pointGeometry.point.pos.srsName				1	1
position.pointGeometry.point.pos.srsDimension				1	1
position.pointGeometry.point.pos.axisLabels				1	1
position.pointGeometry.point.pos.uomLabels				1	1
position.pointGeometry.point.coordinates		Lat long coordinates		0	1
position.pointGeometry.point.coordinates.decimal				0	1
position.pointGeometry.point.coordinates.cs				0	1

position.pointGeometry.point.coordinates.ts				0	1
position: positionGeometryMethod	The manner how this point was defined	PositionGeometry-MethodValueType		1	1
position: positionSpecification	The object on which the point was defined	PositionSpecification-ValueType		1	1
sortfield	Transformation of the house number and the box number (eg. By adding extra 0's before) so this value can be sorted	CharStringType	55	0	1
addressStatus		addressStatusType		1	1
addressStatus.status	Status of the address	AddressStatusValueType	10	1	1
addressStatus.validFrom	Begin date of the validity of the object	dateTime		1	1
addressStatus.validTo	End date of the validity of the object	dateTime		0	1
boxNumber	The box number associated to the address, if any.	CharStringType	35	0	1
houseNumber	The house number associated to the address	CharStringType	15	1	1
officiallyAssigned	Declares if the address is officialy granted (True/False)	Boolean		1	1
hasStreetname		LinkType		1	1
hasStreetname.namespace	Namespace of the street	CharStringType	55	1	1
hasStreetname.objectIdentifier	ObjectIdentifier of the street	CharStringType	10	1	1
hasStreetname.versionIdentifier	VersionIdentifier of the street	CharStringType	20	1	1
hasMunicipality		LinkType		1	1
hasMunicipality.namespace	Namespace of the Municipality	CharStringType	55	1	1
hasMunicipality.objectIdentifier	objectIdentifier of the Municipality	CharStringType	10	1	1
hasMunicipality.versionIdentifier	versionIdentifier of the Municipality	CharStringType	20	1	1
hasPostalInfo		LinkType		1	1
hasPostalInfo.namespace	Namespace of the PostalInfo	CharStringType	55	1	1
hasPostalInfo.objectIdentifier	ObjectIdentifier of the PostalInfo	CharStringType	10	1	1

hasPostalInfo.versionIdentifier	VersionIdentifier of the PostalInfo	CharStringType	20	1	1
isAssignedToAddressableObject	BeSt-Identifier of the addressable object such as buidlingunitId or ParcelId	LinkType		0	n
isAssignedToAddressableObject.namespace	Namespace of the addressable object	CharStringType	55	1	1
isAssignedToAddressableObject.objectIdentifier	ObjectIdentifier of the addressable object	CharStringType	10	1	1
isAssignedToAddressableObject.versionIdentifier	VersionIdentifier of the addressable object	CharStringType	20	1	1
hasPartOfMunicipality		LinkType			
hasPartOfMunicipality.namespace	Namespace of the part-of-mun	CharStringType	55	1	1
hasPartOfMunicipality.objectIdentifier	objectIdentifier of the part-of-mun	CharStringType	10	1	1
hasPartOfMunicipality.versionIdentifier	versionIdentifier of the part-of-mun	CharStringType	20	1	1
beginLifeSpanVersion	date and time at which this version of the object was inserted or changed in the database	dateTime		1	1
endLifeSpanVersion	date and time at which this version of the object was superseded or retired in the database.	dateTime		0	1
event					
event.type	The event that cause the mutation to happen	CharStringType	30	0	1
event.date	The Date the event occurred	Date		1	1
predecessor	Contains the data of the record that will be replaced This will only be filled in on the 'Add' element	Linktype			1
predecessor.namespace	Namespace of the address	Charstringtype	55	1	1
predecessor.objectidentifier	Objectidentifier of the address	CharStringType	10	1	1
predecessor.versionIdentifier	Versionidentifier of the address	CharStringType	20	1	1
successor	Contains the data of the record that is active/latest This will only be filled in on the update element	linktype			1
successor.namespace	Namespace of the address	Charstringtype	55	1	1
successor.objectidentifier	Objectidentifier oof the address	CharStringType	10	1	1
successor.versionIdentifier	Versionidentifier of the address	CharStringType	20	1	1

2.3.4.2 Add & update streetName

Field	Description	Type	Max Length	Min. Occurs	Max. Occurs
homonymAddition	Used for making a unique name , when in large cities a Streetname is used more then once due to merges of municipalities (e.g Atnverpiastraat in Atnwerpen)	CharStringtype	25	0	1
code		IdenifierType		1	1
code.namespace	Namespace of the street	CharStringtype	55	1	1
code.objectIdentifier	Objectidentifier of the street	CharStringtype	10	1	1
code.versionIdentifier	Versionidentifier of the street	CharStringtype	20	1	1
name		GeographicalNameType		1	N
name.spelling	Name of the street	CharStringtype	100	1	1
name.language	Language	LanguageCodeValueType	5	1	1
streetNameStatus		streetnameStatusType		1	1
streetNameStatus.status	The status of the street	streetnameStatusvalueType	10	1	1
streetNameStatus.validFrom	Begin date of the status	dateTime		1	1
streetNameStatus.validTo	End date of the status	dateTime		0	1
type	Type of street	streetNameTypeValueType		1	1
isAssignedByMunicipality		LinkType		0	1
isAssignedByMunicipality.namespace	Namespace of the municipality	CharStringtype	55	1	1
isAssignedByMunicipality.objectIdentifier	objectIdentifier of the Municipality	CharStringtype	10	1	1
isAssignedByMunicipality.versionIdentifier	versionIdentifier of the Municipality	CharStringtype	20	1	1
isAssignedTo		Link Type			
isAssignedTo.RoadObject		RoadObject (LinkType)		0	N
isAssignedTo.RoadObject.namespace	Not used				
isAssignedTo.RoadObject.objectIdentifier	Not used				
isAssignedTo.RoadObject.versionIdentifier	Not used				
isAssignedTo: streetSide	Not used	SideCodeValueType		0	N

BeginLifeSpanVersion	date and time at which this version of the object was inserted or changed in the database	dateTime		1	1
EndLifeSpanVersion	date and time at which this version of the object was superseded or retired in the database.	dateTime		0	1
event					
event.type	The event that cause the mutation to happen	Charstringtype	30	0	1
event.date	The Date the event occurred	DateTime		1	1
Predecessor	Contains the data of the record that will be replaced. This will only be filled in on the ‘Add’ element	IdentifierType		0	N
predecessor.namespace	Namespace of the Streetname	Charstringtype	55	1	1
predecessor.objectidentifier	Objectidentifier of the Streetname	Charstringtype	10	1	1
predecessor.versionId	Versionidentifier of the Streetname	CharStringType	20	1	1
Successor	Contains the data of the record that is active/latest This will only be filled in on the update element	dentifierType		0	N
successor.namespace	Namespace of the Streetname	Charstringtype	55	1	1
successor.objectidentifier	Objectidentifier of the Streetname	CharStringType	10	1	1
successor.versionId	Versionidentifier of the Streetname	CharStringType	20	1	1

2.3.4.3 Add & update municipality

Field	Description	Type	Max Length	Min. Occurs	Max. Occurs
Code		IdentifierType		1	1
code.namespace	namespace of the municipality	charStringType	55	1	1
code.objectIdentifier	objectIdentifier of the municipality (NIS code)	charStringType	10	1	1
code.versionIdentifier	versionIdentifier of the municipality	charStringType	20	1	1
Name		GeographicalNameType		0	N
name.language	Language of the municipality	LanguageCodeValueType	5	1	1
name.spelling	The municipality name	String	100	1	1
municipalityStatus		municipalityStatusType		1	1
municipalityStatus.status	The status of the municipality	municipalityStatusvalueType	10	1	1
municipalityStatus.validFrom	Begin date of the status	dateTime		1	1
municipalityStatus.validTo	End date of the status	dateTime		0	1
Event					
event.type	The event that cause the mutation to happen	CharStringType	30	0	1
event.date	The Date the event occurred	Date		1	1
Predecessor	Contains the data of the record that will be replaced. This will only be filled in on the 'Add' element	Identifiertype		0	N
predecessor.namespace	Namespace of the Municipality	Charstringtype	55	1	1
predecessor.objectIdentifier	Objectidentifier of the Municipality	CharStringType	10	1	1
Ppedecessor.versionIdentifier	Versionidentifier of the Municipality	CharStringType	20	1	1
Successor	Contains the data of the record that is active/latest This will only be filled in on the update element	Identifiertype		0	N
successor.namespace	Namespace of the Municipality	Charstringtype	55	1	1
successor.objectIdentifier	Objectidentifier of the Municipality	CharStringType	10	1	1
successor.versionIdentifier	Versionidentifier of the Municipality	CharStringType	20	1	1

2.3.4.4 Add & update partOfMunicipality

Attribute	Description	Type	Max Length	Min. Occurs	Max. Occurs
code		IdentifierSearchType		1	1
code.namespace	Namespace of the partOfMunicipalityCode. Assigned per region.	NameSpace	55	1	1
code.objectIdentifier	The objectIdentifier of the partOfMunicipality.	String	10	1	1
code.versionIdentifier	The version Identifier of the partOfMunicipalityCode	String	20	1	1
name		GeographicalNameSearchType			
name.spelling	Spelling of the part-of-mun (or part of)	String	100	1	1
name.language	Language of the part-of-mun: Dutch, French or German	LanguageCodeValueType	5	1	1
partOfMunicipalityStatus		streetnameStatusType		1	1
partOfMunicipalityStatus.status	The status of the PartOfMunicipality	streetnameStatusvalueType	10	1	1
partOfMunicipalityStatus.validFrom	Begin date of the status	dateTime		1	1
partOfMunicipalityStatus.validTo	End date of the status	dateTime		0	1
Event				0	1
event.type	The event that cause the mutation to happen	CharStringType	30	0	1
event.date	The Date the event occurred	Date		1	1
Predecessor	Contains the data of the record that will be replaced. This will only be filled in on the 'Add' element	Identifiertype		0	N
predecessor.namespace	Namespace of the PartOfMunicipality	Charstringtype	55	1	1
predecessor.objectIdentifier	Objectidentifier of the PartOfMunicipality	CharStringType	10	1	1

predecessor.versionIdentifier	Versionidentifier of the PartOfMunicipality	CharStringType	20	1	1
Successor	Contains the data of the record that is active/latest This will only be filled in on the update element	Identifiertype		0	N
successor.namespace	Namespace of the PartOfMunicipality	Charstringtype	55	1	1
successor.objectidentifier	Objectidentifier of the PartOfMunicipality	CharStringType	10	1	1
successor.versionIdentifier	Versionidentifier of the PartOfMunicipality	CharStringType	20	1	1

2.3.4.5 Add & update postalInfo

Attirbute	**Description**	**Type**	Max Length	**Min. Occurs**	**Max. Occurs**
code		IdentifierSearchType		1	1
code.namespace	NameSpace of the postcode	NameSpace	55	1	1
code.objectIdentifier	objectIdentifier of the postcode	String	10	1	1
code.versionIdentifier	versionIdentifier of the postcode	String	20	1	1
name		GeographicalNameSearchType			
name.spelling	Spelling of the part-of-mun (or part of)	String	255	1	1
name.language	Language of the part-of-mun: Dutch, French or German	LanguageCodeValueType	5	1	1
postalInfoStatus		streetnameStatusType		1	1
postalInfoStatus.Status	The status of the postalinfo	streetnameStatusvalueType	10	1	1
postalInfoStatus.validFrom	Begin date of the status	dateTime		1	1
postalInfoStatus.validTo	End date of the status	dateTime		0	1

Event				0	1
event.type	The event that cause the mutation to happen	CharStringType	30	0	1
event.date	The Date the event occurred	Date		1	1
Predecessor	Contains the data of the record that will be replaced. This will only be filled in on the ‘Add’ element	Identifiertype		0	N
predecessor.namespace	Namespace of the postName	Charstringtype	55	1	1
predecessor.objectIdentifier	Objectidentifier of the postName	CharStringType	10	1	1
predecessor.versionIdentifier	Versionidentifier of the postName	CharStringType	20	1	1
Successor	Contains the data of the record that is active/latest This will only be filled in on the update element	Identifiertype		0	N
successor.namespace	Namespace of the postName	Charstringtype	55	1	1
successor.objectIdentifier	Objectidentifier oof the postName	CharStringType	10	1	1
successor.versionIdentifier	Versionidentifier of the postName	CharStringType	20	1	1

2.3.5 Processing mutations

File naming conventions

The naming conventions are analogous to these for the Full Download. For each BeSt object, each Region will deliver 4 zipped files (and Wallonia 5):

<u>Region files</u>

The same naming convention as for the full download applies.

<u>BOSA files</u>

The naming convention for BOSA's zipfile is

“BeStAddress_” + UseCaseName (abbreviation) + ‘Belgium’ + Date

Mutations: example: BeStAddress_MBelgium20200312.zip

Processing sequence

When processing mutations, the correct sequence has to be observed.

For referential integrity reasons (during for the mutation processing), the XML is built up in a particular order.

First the Add elements

1. Municipality
2. PartOfMunicipality
3. PostalInfo
4. StreetName
5. Address

Then the UPD elements

1. Address
2. StreetName
3. PostalInfo
4. PartOfMunicipality
5. Municipality

In no case, the processing of a record can introduce a referential error by referring to a record that still must be added.

Examples:

- A new address is added in a new street. So, first the new street has to be processed before the new addresses can ber processed
- When a Municipality is replaced by another.
  You first replace all Addresses belonging to the replaced Municipality (and Streetname) with the Addresses linked to the current Municipality (and Streetname), After this is done, you do the same for the Streetnames before you replace the Municipality itself.

An “Add” record contains the predecessor (if it exists). This indicates which record was it's predecessor.

The “Update” records contains predecessors and successors. This allows to construct a historty chain.

Example: A StreetName gets a name correction. StreetName with identifier “1” version “1” is corrected into a new version with identifier “1” and version ”2”.

The following data (schematic) mutations are provided:

Record type	Object Identifier	Version Identifier	Valid From	Valid To	Event type	Prede cessor	Suc cessor
Add	1	2	1/5/2022	-	Correction	1.1	
Update	1	1	1/1/2000	1/5/2022			1.2

Possible error situations

To be prepared.

2.3.6 Results in mutations file by type of change

This section explains how the different types of changes (new address, update of an address, new address component, update of address components) will result in mutations records.

Examples include fields filled with possible values.

Conventions

1. In case a new object is created

    - The term “previous” refers to the initial object that will become inactive after the mutations are processed
    - The term “next” refers to the object that will replace the previous object and will become active after the mutations are processed

2. Component

    As defined in the glossary, this term is used to refer to a sub-part of an Address or streetName.

    The following are all address components:

    - Municipality
    - partOfMunicipality
    - postalInfo
    - StreetName

    StreetName only has one component, i.e the municipality

2.3.6.1 New address

When a municipality has a need for a new address to identify a location (buildingunit, parcel, moorplace, ….), a new address will be created without any link to a previous address.

The possible reason for creating new addresses is that the municipality wants to link an addressable object (building unit or parcel) an address and since that building unit or parcel is new, the address is new as well

Examples:

- a new parcel is created
- a number of building units are destroyed and are replaced by other building units. These are also a new addresses with no links to previous building units

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	All data fields are filled in	Current, Reserved or Proposed	Date provided by region			Date provided by Region		

2.3.6.2 Change to address fields houseNumber, BoxNumber, geo coordinates or status

When a change to any of these fields is made, the BeStidentifier of the address changes, which indicates that a replacement of an address has happened. The application will provide following mutations:

- an upd element for the previous address stating this one is no longer valid and a link to the successor.
- an add element for the next address including the predecessor with the link to the previous address.

This makes it possible to replace the previous address with the next address in all your systems.

Expected result

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
add	All data fields filled in	Current	Date provided by regions			Date provided by regions	Previous address BeStId	
upd	All fields filled in	Current	Date provided by regions	Date provided by regions		Date provided by regions		Next address BeStId

Alternative results of the changes to address field(s) (Flanders region)

1. Due to asynchronous handling of events in Flanders it is possible that the link between previous address' BeSt id and the next address' BeStid is not yet known. Is this case, the predecessor and successor information will not be sent at the time the change happened but updates will be sent with a delay of 1 day.

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
add	All data fields filled in	Current	Date provided by regions			Date provided by regions		
upd	All data fields filled in	Current	Date provided by regions	Date provided by regions		Date provided by regions		

The next day, the mutations file will include following records:

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
upd	All data fields filled in	Current	Date provided by regions			Date provided by regions	Previous address BeStId	
upd	All data fields filled in	Current	Date provided by regions	Date provided by regions		Date provided by regions		Next address BeStId

2. Due to missing information in the Flanders application about links between physical locations of mooring places and trailer parks, BeSt. will not yet be able to provide the predecessor and successor for these. A solution remains to be worked out with Flanders.

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
add	All data fields filled in	Current	Date provided by regions			Date provided by regions		
upd	All data fields filled in	Current	Date provided by regions	Date provided by regions		Date provided by regions		

2.3.6.3 Administrative corrections to address fields houseNumber, BoxNumber or geo coordinates

In case of an administrative correction, the person who made the change identified that change as a “correction” (= special type of change in the applications of the municipality).
When an administrative correction to any of these fields is made, the BeStidentifier of the address changes.

The application will provide following mutations:

- an upd element for the previous, incorrect address stating this one is no longer valid and a link to the successor.
- an add element for the corrected address including the predecessor with the link to the previous address.

To distinguish such “correction” from a “true change”, BeSt will set eventtype = ‘CORRECTION’ in the add element.

Expected result:

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
add	All fields filled in	Current	Date provided by regions		CORRECTION	Current date	Previous address BeStid	
upd	All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		Next address BeStid

2.3.6.4 New address component: Streetname, Municipality, PostalInfo or PartOfMunicipality

When an address component is created without any link to previously known components this is defined as a new address component.

Expected results for the address component:

element Name	Element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
add	All data fields are filled in	Current, Reserved or Proposed	Date provided by region			Date provided by Region		

2.3.6.5 Change to the value of a field of an AddressComponent

This section applies for all addressComponents. Below, an examle is given for the addressComponent streetName.
When a change is made to an AddressComponent field (e.g. to a field of the adress component streetName), the BeStidentifier of the addressComponent changes.
The BeSt application will provide following mutations (example given for a change to the name of a street):

- an upd element for the previous streetName stating that it is no longer valid (validTo date filled in) and a link to the successor.
- an add element for the next streetName including the predecessor with the link to the previous StreetName

Expected result for the addressComponent (streetName)

element name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	All fields filled in	Current	Date provided by regions			Current date	Previous streetname	
Upd	All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		Next streetname

This makes it possible to replace the previous streetName with the next streetName in all your systems.

Note:
The change to an addres-component field can be to a field known to BeSt or to another field in the region application that is *unknown* to BeSt. In the latter case, the result will be that there is an Upd of the previous record and an Add of the new record as above; however, the Add will only have a new BeStidentifier and all other fields remaing unchanged. This can happen to every address component.

Expected result for the affected addresses

Since the BeStIdentifier of the streetName changes, the streetName BeStIdentifier in all addresses of that street will be updated. The resulting mutations for addresses are described in section 0 for streetName/postalinfo and partOfMunicipality changes and in section 0 for municipality changes

2.3.6.6 Replacement of an AddressComponent

When an AddressComponent is replaced due to a name change, the BeStidentifier of that AddressComponent changes.

The BeSt application will provide the same mutations as described in the previous paragraph.

2.3.6.7 Administrative correction to an AddressComponent

When a correction is made to the name of an address component, the BeStidentifier of that address component changes.

The BeSt application will provide following mutations:

- an upd element for the previous address component stating that it is no longer valid (validTo date filled in) and a link to the successor.
- an add element for the next address component including the predecessor with the link to the previous address component

To distinguish such "correction" from a "true change", BeSt will set eventtype = 'CORRECTION' in the add element.
In some cases, no new version is made of the record, then the consumer only receives an Upd for the address.

The expected result for the ***Address component***:

element Name	Element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	All fields filled in	Current	Date provided by regions		CORRECTION	Current date	Previous component	
Upd	All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		Next component

Impact on *addresses* linked to the address component

Since the BeStIdentifier of the streetName changes, the streetName BeStIdentifier in all addresses of that street will be updated. The resulting mutations for addresses are described in section 0 for streetName/postalinfo and partOfMunicipality changes and in section 0 for municipality changes

2.3.6.8 Changes to an address due to a change to its Streetname or PostalInfo or partOfMunicipality component

Any change that happens to a component element (other than municipality) will result in a change of the BeStidentifier of that component and in a change of the BeStidentifier of the address records that are linked to the changed component. The result is a replacement of these addresses.

Expected result:

element Name	**element**	**Status**	**ValidFrom**	**ValidTo**	**EventType**	**EventDate**	**Predecessor**	**successor**
Add	All fields filled in	Current	Date provided by regions			Current date	Previous address	
Upd	All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		Next address

2.3.6.9 Changes to Address due to a change to its municipality component

In case of a change to the municipality component, there are 2 possible outcomes for the addresses linked to that municipality:

1. either there is an Add and an Upd as described in the previous paragraph
2. or there is *only* an Upd due to the fact that the BeStidentifier of the address doesn't change but only the versionId of the reference to the municipality in the address (Flanders)

1. The BeSt identifier of the address changes

Expected result for the affected addresses:

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	All fields filled in	Current	Date provided by regions			Current date	Previous address	
Upd	All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		Next address

2. The BeStidentifier of the address doesn't change.

When a Municipality name has been corrected in Flanders, this will lead to a version update of the Municipality, but without new BeStidentifier for the addresses linked to that Municipality. (reason being that Flanders, internally in their system, does not use the versionIdentifier of the Municipality at Address level)

Expected result for the affected addresses:

The municipality BeStidentifier has changed and thus all addresses of that municipality will receive an update; however, their BeSt identifier will not change

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Upd	All fields filled in	Current	Date provided by regions			Current date		

2.3.6.10 Changes to streetName due to a change to its municipality component

The logic applied here is analogous to the one for addresses in the previous paragraph.
In case of a change to the municipality component, there are 2 possible outcomes for the streets linked to that municipality:

1. either there is an Add *and* an Upd
2. or there is *only* an Upd due to the fact that the BeStidentifier of the streetName doesn't change but only the versionId of the reference to the municipality in the streetName (Flanders)

1. The BeSt-identifier of the streetName changes

<u>The expected result for the affected streetname elements:</u>

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	All fields filled in	Current	Date provided by regions			Current date	Previous Streetname	
Upd	All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		Next Streetname

2. The BeStidentifier of the streetname doesn't change.

The municipality BeStidentifier has changed and thus all streetNames of that municipality will receive an update; however, their BeSt identifier will not change.

<u>The expected result for the affected streetname elements:</u>

element Name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Upd	All fields filled in	Current	Date provided by regions			Current date		

2.3.6.11 Split of an AddressComponent of format A = A+B

When an AddressComponent is split as A = A+B, the BeSt application will provide following mutations:

- an upd element for the AddressComponent A stating there is a successor (B); nothing else is updated since the AddressComponent A continues to exist
- an add element for the new AddressComponent B including a reference to the predecessor (A).

Expected result for the address component

element name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	B : All fields filled in	Current	Date provided by regions			Current date	A	
Upd	A : All fields filled in	Current	Date provided by regions			Current date		B

Impact on addresses linked to the address component
For the addresses that remain part of address component A, there is no change as the BeStIdentifier of A remains the same.

However, the new address component B has a new BeStIdentifier.
Since the BeStIdentifier of the address component B is new, the address component's BeStIdentifier in all addresses linked to address component B will be updated.
The resulting mutations for addresses are described in section 0 for streetName/postalinfo and partOfMunicipality changes and in section 0 for municipality changes

2.3.6.12 Split of an AddressComponent of format D = E+F

In this case, the initial address component ceases to exist after the split and is split into 2 new address components.
When an AddressComponent is split as D = E+F, the BeSt application will provide following mutations:

- an upd for the AddressComponent D stating there are successors (E,F) and the validTo date is filled in, stating this address component D is no longer valid.
- add elements for the new AddressComponents E and F including a reference to the predecessor (D).

The expected result for the address components:

element name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	E : All fields filled in	Current	Date provided by regions			Current date	D	
Add	F : All fields filled in	Current	Date provided by regions			Current date	D	
Upd	D : All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		E,F

Impact on addresses previously linked to the address component D
Since the BeStIdentifier changes for all addresses previously belonging to AddressComponent D, these addresses will receive an update of their AddressComponent BeStIdentifier
The resulting mutations for addresses are described in section 0 for streetName/postalinfo and partOfMunicipality changes and in section 0 for municipality changes

2.3.6.13 Merge of an AddressComponent of format K+L = K

When a AddressComponent is Merge as K+L = K, the BeSt application will provide following mutations:

- an upd element in for the AddressComponent L stating there is a successor (K), the validTo date of L is filled in, stating that L is no longer valid
- an upd element for the AddressComponent K including a reference to the predecessor (L)

The expected result for the address components:

element name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Upd	L : All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		K
Upd	K : All fields filled in	Current	Date provided by regions			Current date	L	

Impact on addresses previously linked to the address component L:
Since the BeStIdentifier changes for a part of the addresses (the addresses of L that has now become a part of K), these addresses will receive an update of their AddressComponent BeStIdentifier.

The resulting mutations for addresses are described in section 0 for streetName/postalinfo and partOfMunicipality changes and in section 0 for municipality changes

2.3.6.14 Merge of an AddressComponent of format P+Q = R

In this case, two initial address component cease to exist after they are merged into one new address component.

When a AddressComponent is merged as follows P+Q=R, the BeSt application will provide following mutations:.

- upd elements for the AddressComponents P and Q stating they now have a successor (R), the validTo date for P and Q is filled in, stating they are no longer valid.
- an add element for the new AddressComponent R , including references to its predecessors(P,Q).

Expected result for the address components:

element name	element	Status	ValidFrom	ValidTo	EventType	EventDate	Predecessor	successor
Add	R : All fields filled in	Current	Date provided by regions			Current date	P,Q	
Upd	P : All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		R
Upd	Q : All fields filled in	Current	Date provided by regions	Date provided by regions		Current date		R

Impact on addresses previously linked to the address components P and Q

Since the BeStIdentifier changes for addresses previously linked to AddressComponent P and Q, these addresses will receive an update of their AddressComponent BeStIdentifier.

The resulting mutations for addresses are described in section 0 for streetName/postalinfo and partOfMunicipality changes and in section 0 for municipality changes

2.4 SXXX - StructuralAnomalyFileService

Main functionality

BOSA will perform a number of validations on incoming address data from the regions to verify completeness and consistancy. These validations may produce a list of “structural” anomalies, that BOSA will report to the regions.

Most of these anomalies are found when Bosa compares the full download with the BoSa database (weekly action). But also the BoSa processing of mutations can detect anomalies.

All the anomalies for 1 region are collected in 1 weekly “structural anomalies file” that is send to the concerned region. These files will be available to download by any interested party.

Types of anomalies

- “Missing element”
    - ➢ An element is present in the full download but not in BoSa DB
- ❖ “Remaining active” element
    - ➢ An element is still active according to the BoSa DB, but not present as such in the full download.
- ❖ “Delta element”
    - ➢ The element is present in both systems (Full download and database)
    - ➢ The content does not correspond.
        - ▪ This test includes all fields, independent from the status.
- ❖ “Double active” element
    - ➢ Element with at least 2 versions that are not closed (no ValidTo date filled in)
- ❖ “Invalid reference” (Reference is pointing to element that is missing in the database)
    - ➢ Address
        - ▪ Streetname invalid
        - ▪ Postinfo invalid
        - ▪ Municpality invalid
        - ▪ Part of municipality invalid
        - ▪ Predecessor invalid
        - ▪ Successor invalid
    - ➢ StreetName
        - ▪ Assigned By Municipality invalid
        - ▪ Predecessor invalid
        - ▪ Successor invalid
    - ➢ Municipality
        - ▪ Predecessor invalid
        - ▪ Successor invalid
    - ➢ PostalInfo
        - ▪ Predecessor invalid
        - ▪ Successor invalid
    - ➢ PartOfMunicipality
        - ▪ Predecessor invalid
        - ▪ Successor invalid
- ❖ “Double history”
    - ➢ Address with more than 1 predecessor / successor
- ❖ “Bad location”
    - ➢ Address with empty or “0-0” position
- ❖ “Not De-activated”
    - ➢ Active Address located in non-active street / municipality
        - ▪ For Postcode and Part of municipality, this check is currently impossible because there is no status available.
        - ▪ For Municipality, Bosa keeps a list of municiplaities that did merge in the past.
    - ➢ Active Street in a non-active municipality
        - ▪ See additional list of merged municipalities

- “Confusing municipality”
  - An address and it’s street points to different municipalities
    - Address:
      - has Municipality
      - has Streetname
    - Streetname
      - Assigned by municipality
- Status Replacement
  - A record is found where the status did change within this same version.
- Unknown namespaces
  - Verify that the namespace exists in the list of accepted namespaces.
- Incorrect enumeration
  - Certain fields should have a certain value. We accept anything, so this must be checked.
- Incorrect TimeLine
  - ValidFrom > ValidTo or BeginlifeSpanVersion > EndLifeSpanVersion (if filled)

File layout

The structure of the file:

- elementType
  - Enumeration of possible elements:
    - “M” Municipality
    - “O” Part Of Municipality
    - “P” PostalInfo
    - “S” streetName
    - “A” address
- AnomalyCode
  - Integer value, identifying the anomaly.
- AnomalyName
  - The name of the anomaly as it is introduced in types of anomalies
- Region
  - Enumeration:
    - F: Flanders
    - B: Brussels
    - W: Wallonia
- Municipality
  - The postcode or Niscode of the municipality where this error is found
- BeStId
  - BeStId of the anomaly record
  - Combination of namespace / objectIDentifier / versionIdentifier
- additionalParameters
  - Many anomalies have specific attributes. Not every anomaly needs the same attributes to explain the anomaly. We try to inform the receiver as good as possible about all information concerning the anomaly.

Additional information of the anomalies

- “Missing element”
  - AnomalyCode: 1
  - AdditionalParameters: none
- “Delta element”

    - AnomalyCode: 2
    - AdditionalParameters: List of differences, structure [name;value full download, value DB]
- “Double active” element
    - AnomalyCode: 3
    - AdditionalParameters: List of versions
- “Double history”
    - AnomalyCode: 4
    - AdditionalParameters: List of allpredecessors / successors
- “Bad location”
    - AnomalyCode: 5
    - AdditionalParameters: Coordinates
- “Invalid reference” (Reference is pointing to element that is missing in the database)
    - AnomalyCode: 6
    - AdditionalParameters: The bad reference: reference Name – Reference (e.g. ‘Municiplaity’ – XXXX/YYY/ZZZ)
- “Not De-activated”
    - AnomalyCode: 7
    - AdditionalParameters: reference name – Status + validity preriod current object – status + validity period referenced element
- “Confusing municipality”
    - AnomalyCode: 8
    - AdditionalParameters: Reference to municipality in Address - Reference in StreetName
- “Remaining active” element
    - AnomalyCode: 9
    - AdditionalParameters: none
- Status Replacement
    - AnomalyCode:10
    - AdditionalParameters: new Status, old Status
- Unknown namespaces
    - AnomalyCode: 11
    - AdditionalParameters: name field, status value
- Incorrect enumeration
    - AnomalyCode: 12
    - AdditionalParameters: name field, incorrect value
- Incorrect TimeLine
    - AnomalyCode: 13
    - AdditionalParameters: name field – value from – value to

File naming conventions

Similar rules apply as for the Full Download. For each service, each Region will deliver 4 zipped files (and Wallonia 5):

Region files

The same naming convention as for the full download applies.

BOSA files

The naming convention for BOSA's zipfile is

“BeStAddress_” + UseCaseName (abbreviation) + ‘Belgium’ + Date

Mutations:_ example: BeStAddress_**M**Belgium20200312.zip

Errors

Whenever there is an error concerning the MFT services, it would result in a missing file for a certain region.
This could be a one-time event or a recurring error.
In every case, the client will only see that there are files missing.
If this is the case, BOSA will contact the responsible region(s) to solve the problem.

3 Open Data

3.1 Full download files

A copy of the latest MFT “Full Download” ZIP file is published every Sunday morning on the static website opendata.bosa.be website as https://opendata.bosa.be/download/BeSt/BeSt-full-latest.zip .

In addition, a version in CSV and some derived lists are also offered to the public, but without any guarantee.

This data can be used by anyone (including companies / entities with no affiliation with BOSA whatsoever) , and without any registration.

3.2 Mutation files

Daily mutation files are not copied to the open data site.

4 Known issues

4.1 Flanders region

Due to missing information in the Flanders application about links between physical locations of mooring places and trailer parks, BeSt. will not yet be able to provide the predecessor and successor for these at this time. A solution remains to be worked out with Flanders.

Document Information

General

Authors(s) :	Gert De Jonge, Stijn Adriaenssens, Eddy Corthouts
Document name:	BeStAddress Userguide MFT services
Location of the document:	
Number of pages:	58
Version:	2.6
Print date:	2022/12/15

Approbation

Nom	Fonction	Organisation
Johan Mertens	Service Manager	BOSA
François Soumillion	Integration Architect	BOSA

Distribution

This document will be distributed to:

Name	Function	Organisation	Objective of distribution