aSISt

Reporting service system

Advanced User Guide

version 5.40.6.0

Wroclaw, October 2020

Table of contents

Table of Figures

1 Introduction

aSISt, tranSIS and aSISt server are a comprehensive solution for data institutions for obligatory reporting purposes, as well as for data gathering and analysis by the supervising entities.

aSISt is designed for entities bound to prepare periodic reports for the supervisor. Basic aSISt functions enable entering, verification and delivery of data to the supervisor.

tranSIS enables the supervisor to gather reports, created in aSISt or another XBRL-compliant application, from single institutions. With tranSIS, the supervisor can verify and analyze reported data, as well as define their range.

axSIS can be used by the reporting institution do analyze its own reports, as well as reports of other entities. The main purpose of this application is to enable easy data import and analysis.

aSISt server is designed to remotely perform time-consuming operations which require processing large volumes of data.

1.1 aSISt

aSISt enables report preparation in obligatory reporting processes. The range and form of data entered is defined in a taxonomy – a set of XML Schema and XML files, compliant with XBRL 2.1, XBRL Dimension 1.0 and XBRL Formula standards.

All reporting types have their own taxonomy which defines the range of reported data, form of presentation and data validation control rules. Partly, taxonomies were created on the basis of visual reference materials delivered by a supervisor, as well as normative documents and official electronic resources. A particular case are COREP and FINREP taxonomies, created by an international organization CEBS, adjusted to the Polish reporting system by the National Bank of Poland (NBP) and made available directly as an XBRL taxonomy. NBP created also an XBRL taxonomy for NB300 and LE reports. In these taxonomies, the XBRL Formula standard is used to validate a report.

Basic aSISt functions are: providing data entry mechanisms, data verification and final report preparation (in a format defined by the supervisory entity).

Data in aSISt can be entered manually, imported from external files (CSV, Excel, XML, etc.) or imported with a flexible mappings module which allows for data feeding to tabular sets (CSV, Excel, SQL databases, etc.), as well as other, e.g. hierarchical sets (for example XML files). The mappings module is described in details in the following sections of this document.

Correctness of data entered can be validated with rules which facilitate entering a suitable amount of reported data. Rules can be defined by the supervisor and delivered together with the taxonomy. The reporting entity can also define its own rules using the rules module. These rules ensure correctness of data entered and their conformance with expectations of the reporting entity. The rules module is described in details in the following sections of this document.

Report preparation process results in report generation in a defined by the supervisor format. Currently, aSISt enables output data generation in an XBRL format, among others, and many other formats specific to certain types of reporting. XBRL reports can be created for any supported reporting type. An XBRL report can be processed by any XBRL tool which enables work with instance documents and XBRL taxonomy.

1.2 tranSIS

tranSIS can be used by supervisory institutions to gather and analyze data prepared by entities with periodic reporting obligations. Examples of this process are cooperative banks preparing reports for associating banks.

tranSIS can work with aSISt, therefore supervisory data can be delivered directly from aSISt. To enable this process, a flexible JMS (Java Message Service) mechanism was implemented.

tranSIS users can verify and analyze delivered data. Delivered reports can be viewed with the same mechanism as in aSISt. Additionally, tranSIS users can inform reporting entities on the report validity status through JMS mechanisms.

One of the main tasks of tranSIS is gathering data from subordinate institutions and preparing a consolidated report which can be sent to an appropriate superordinate institution. Currently, tranSIS enables preparation of consolidated reports which are a simple aggregate or preparation of more complex reports, using the XBRL standard (e.g. aggregate instance for COREP and FINREP), or even more complex aggregate instances, created with the mappings module (data is transformed in the process of aggregate report preparation, using expressions from the language of the mappings module).

1.3 aSISt server

aSISt server is designed to remotely perform time-consuming operations which require processing large volumes of data. It receives tasks defined in aSISt, queues them and then processes. Results are sent to aSISt for further work. Communication between the server and the client applications is possible with the SOAP protocol.

1.4 Related documents

- aSISt - user guide.pdf– aSISt User Guide
- aSISt - installation guide.pdf – aSISt Installation Guide

1.5 Terms

- **XBRL** (Extensible Business Reporting Language) – an XML-based language, dedicated to financial reports descriptions and preparation
- **calculations** – arithmetic rules defining relations between values in a report; only addition (of value lists) with weights is supported
- XBRL **instance** – a resulting financial report in XBRL format

- banking **entity** – a single banking unit with assigned reporting data; there are two default units in the system – for individual and consolidated reports; an optional consolidation module enables users to create reports for additional units (banking entities)
- **report** – a set of reporting data for a given reporting period, pertaining to a single taxonomy (COREP, FINREP, WEBIS) and assigned to a specific banking entity.

1.6 General technical information

aSISt, tranSIS, axSIS and aSISt server are Java-based applications and require **Java 8** environment. They operate on free repositories and therefore do not require additional licenses.

aSISt enables data storing in a built-in **Apache Derby** (v10) database, also known as JavaDB. This database, due to its limitations, may be used with single workstation versions only. In multi workstation versions, aSISt allows for **Oracle 9i/10g/11g** database.

tranSIS, due to potentially high data volume, enable data storing in **Oracle 9i/10g/11g** database only.

aSISt server stores operation-critical data in **Apache Derby** database.

Due to a 4GB stored data limit in the Oracle Database Express Edition (Oracle Database XE), users choosing this database must pay particular attention to data volume and monitor the level of remaining free resources.

2 System architecture

aSISt, tranSIS and aSISt server are based on a three-tier architecture, consisting of:

- presentation level,
- servicing level,
- database level.

2.1 aSISt

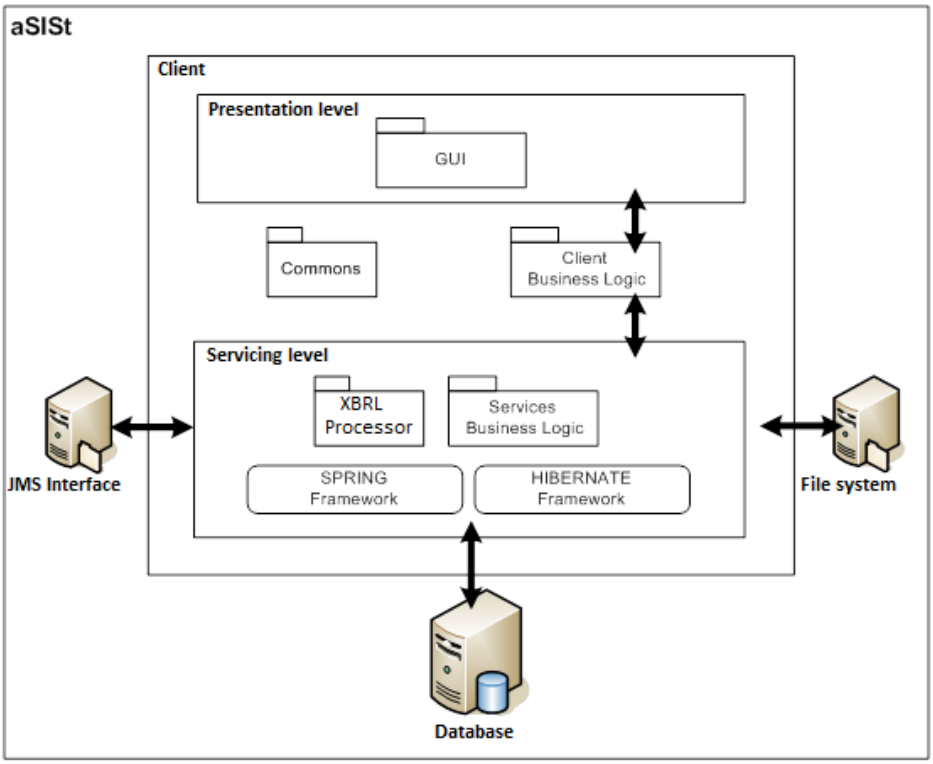


Figure 1. aSISt – system architecture

2.2 tranSIS

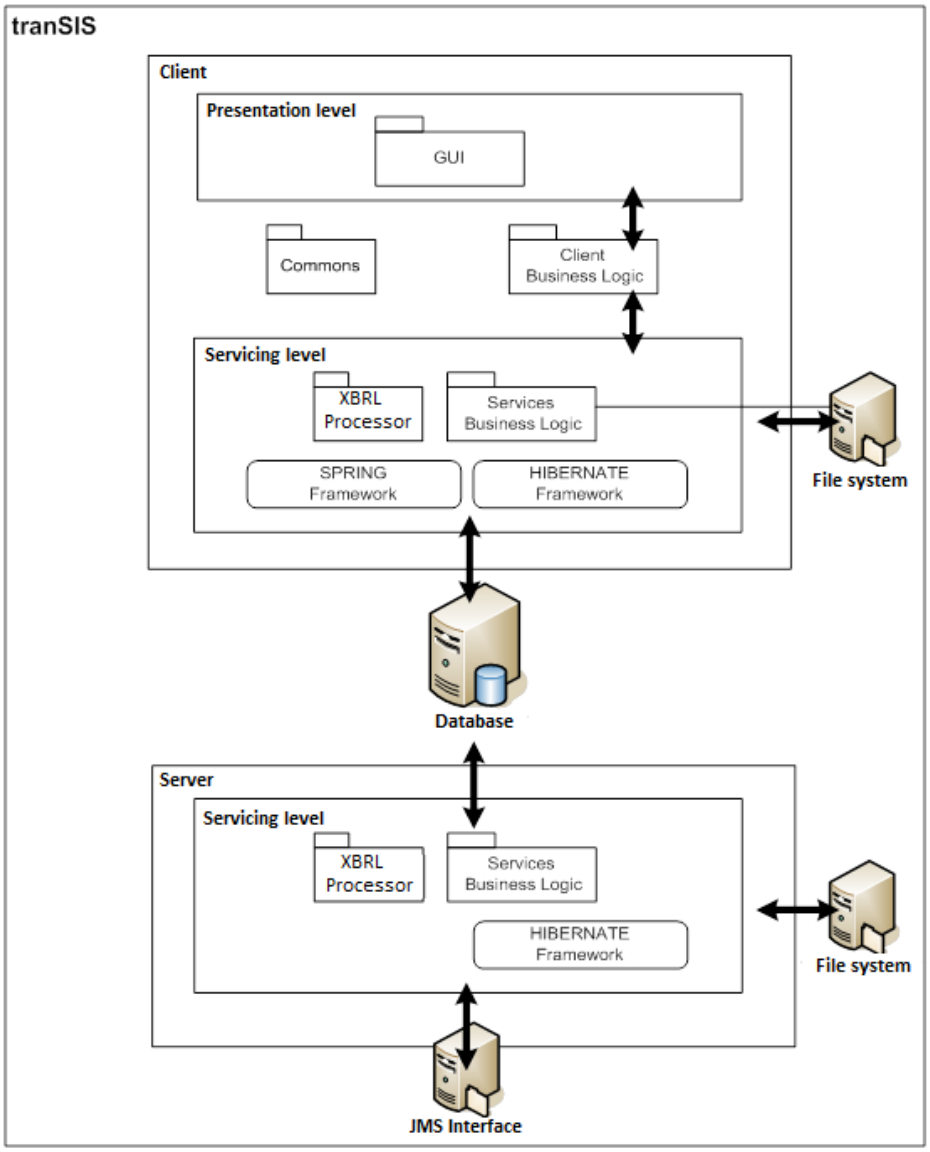


Figure 2. tranSIS – system architecture

aSISt/tranSIS system work in a "fat client" mode – the logic of presentation and service levels is implemented on the client side.

Apart from communication with the database, the client communicates also with the file system, in which selected application configuration data, client data and temporary data are stored. These data are saved locally due to performance requirements.

aSISt and tranSIS can use JMS interface which enables communication between these two applications. aSISt can have direct access do JMS service. tranSIS on the other hand uses the application called tranSIS Server to connect

with JMS service. tranSIS Server receives and sends messages from aSISt. Data processed in this application are moved to a database and from there can be viewed with a client called tranSIS Console.

aSISt consists of two modules:

- processing engine
- managing console

Both modules run as applications of an in-built **Jetty** application server. The first of the two modules, processing engine, enables communication with aSISt clients and performance of commissioned tasks. Managing console was created in order to facilitate supervision of the processing engine performance. It was implemented in HTML, therefore it is accessible from every modern Internet browser.

3 Users and permissions

3.1 Authentication

User authentication is based on a standard system log in process. Logins (user names) in the system are unique. Passwords are encrypted with hash function (MD5).

3.2 Authorization

Authorization is based on roles. Each user can be assigned any set of rules.

A role is a set of permissions determining access to the system functions.

Set of rules depends on the system (aSISt/tranSIS) and database type (Derby/Oracle). Permissions have defined ranges (specified relations with taxonomies and banking entities).

3.3 Permissions

- creating periods (***CreatePeriod***) – Generally available, for the entire range.
- changing current period (***SwitchPeriod***) – Generally available, for the entire range.
- generating instances (***GenerateInstance***) – Generally available, for the entire range.
- creating archives (***CreateArchive***) – Available for aSISt only, for the entire range.
- restoring periods from archives (***RollbackArchive***) – Available for aSISt only, for the entire range.
- deleting archives (***RemoveArchive***) – Available for aSISt only, for the entire range.
- changing period statuses and switching to revision mode (***ChangePeriodStatus***) – Generally available, for the entire range.

- importing taxonomies (***ImportTaxonomySet***) – Generally available, for the entire range.
- modifying global settings (***ChangeGlobalSettings***) – Generally available, for the banking entities range.
- managing users: adding, removing, modifying (***ManageUsers***) – Generally available, for the banking entities range.
- modifying own user data (***ModifyOwnAccount***) – Generally available, no ranges.
- modifying user roles (***ModifyUserRoles***) – Generally available, for the banking entities range.
- modifying data (***ModifyData***) – Generally available, for the entire range.
- exporting data (***ExportData***) – Generally available, for the entire range.
- importing data (***ImportData***) – Generally available, for the entire range. Permission available together with the (*ModifyData*) permission.
- backing up database (***BackupDatabase****)* – Available for aSISt and single workstation mode only, no ranges.
- restoring database from back up *(**RestoreDatabase***) – Available for aSISt and single workstation mode only, no ranges.
- modifying period settings *(**ModifyPeriodContext***) – Generally available, for the entire range.
- rounding data (***RoundingData****)* – Generally available, for the entire range. Permission available together with the (*ModifyData*) permission.
- accessing activity list *(**ViewActivity***) – Generally available, for the entire range.
- accessing updates *(**UpdateApp****)* – Available for aSISt only, for the entire range.
- modifying data in internal audit mode (***ModifyAuditLevelInternal***) – Generally available, for the entire range.
- modifying data in tranSIS audit mode (***ModifyAuditLevelTransis***) – Generally available, for the entire range.
- modifying data in supervision audit mode (***ModifyAuditLevelSupervision***) – Generally available, for the entire range.

  Audit levels are hierarchy-based:

  - ModifyAuditLevelSupervision
    - ModifyAuditLevelTransis
      - ModifyAuditLevelInternal

- rule management, rule module *(**RuleManagement***) – Generally available, for the entire range.
- bank units' management (***ManageBankUnits****)* – Generally available, for banking entities range.
- managing analytical reports module (***ManageAnalyticalReports***) – Generally available, for the entire range.
- managing analytical reports which belong to a different user (***AnalyticalReportAdmin***) – Generally available, for the entire range.
- managing prospectus module *(**ProspectusAdmin***) – Generally available, for the entire range.
- managing mappings module *(**MappingsManage***) – Available for aSISt only, for the entire range.
- reading/importing mappings *(**MappingsRead****)* – Available for aSISt only, for the entire range.
- blocking report *(**BlockReport****)* - Available for aSISt only, for the entire range.
- editing keystore *(**EditKeyStore****)* - Available for aSISt only, for the entire range.
- editing exclusions *(**EditExclusions****)* - Available for aSISt only, for the entire range.
- managing exclusions (***ExclusionsAdmin***) - Available for aSISt only, for the entire range. Permission available together with the (*EditExclusions*) permission.

- editing corrections (***EditCorrections***) - Available for aSISt only, for the entire range.
- managing snaphots *(**DataRepositoryDefinition**)* - Available for aSISt only, for the entire range.
- generating consolidated report (***GenerateConsolidatedReport***) – Available for tranSIS only, for the banking entities range.

3.4 Roles

By default, there are 5 roles available in the application:

- **System Administrator** – has all permissions,
- **Administrator** – has the following permissions:

  'CreatePeriod', 'SwitchPeriod', 'GenerateInstance', 'CreateArchive'(unavailable in tranSIS)**, 'RemoveArchive'**(unavailable in tranSIS)**, 'RollbackArchive'**(unavailable in tranSIS)**, 'ChangePeriodStatus', 'ImportTaxonomySet', 'ChangeGlobalSettings', 'ModifyOwnAccount', 'ModifyData', 'ExportData', 'ImportData', 'BackupDatabase'**(unavailable for Oracle database)**, 'RestoreDatabase'**(unavailable for Oracle database)**, 'ModifyPeriodContext', 'RoundingData', 'ViewActivity', 'RuleManagement', 'UpdateApp'**(unavailable in tranSIS)**, 'ModifyAuditLevelSupervision', 'ManageBankUnits', 'ManageAnalyticalReports', 'AnalyticalReportAdmin', 'MappingsManage'**(unavailable in tranSIS)**, 'MappingsRead'**(unavailable in tranSIS)**, 'TaxonomyScope', 'BankUnitScope', 'BlockReport'** (unavailable in tranSIS) **, 'EditKeyStore'** (unavailable in tranSIS) **, 'EditExclusions'** (unavailable in tranSIS)**, 'ExclusionsAdmin'** (unavailable in tranSIS)**, 'EditCorrections'** (unavailable in tranSIS)**, 'GenerateConsolidatedReport '** (unavailable in aSISt)

- **Operator** – has the following permissions:

  'SwitchPeriod', 'GenerateInstance', 'CreateArchive', 'RemoveArchive', 'RollbackArchive', 'ImportTaxonomySet', 'ModifyOwnAccount', 'ModifyData', 'ExportData', 'ImportData', 'MappingsRead', 'BackupDatabase', 'RoundingData', 'ViewActivity', 'ModifyAuditLevelSupervision', 'ManageAnalyticalReports', 'MappingsManage', 'TaxonomyScope', 'BankUnitScope', 'BlockReport', 'EditExclusions', 'GenerateConsolidatedReport '

- **Analytical Module Operator –** has the following permissions:

  'SwitchPeriod', 'ExportData', 'BackupDatabase', 'ModifyOwnAccount', 'ManageAnalyticalReports', 'AnalyticalReportAdmin', 'TaxonomyScope', 'BankUnitScope'

- **Reader** – has the following permissions:

  'SwitchPeriod', 'ExportData', 'BackupDatabase', 'ModifyOwnAccount', 'TaxonomyScope', 'BankUnitScope'

Role names and types are recorded in the 'Role' table.

Permission set which creates a role is recorded in the 'Permission' table.

Role management is available from the application. It comprises of role creating as well as role assigning to a specific user.

Roles may be modified by the user with 'ModifyUserRoles' permission.

3.5 Users

By default, there is one 'Admin' user available in the application. The 'Admin' user has an assigned **'System Administrator'** role.

4 Work in a multiple-user environment

4.1 Work mode choice

The choice of application work mode is based on the defined **db.type** entry in the **db.properties** file. This entry can have the following values:

- **db.type=derby**: single workstation work with a built-in Apache Derby database (available in aSISt only)
- **db.type=oracle**: work in a network with an Oracle database

4.2 Work in a network mode

With multiple users working simultaneously, it is potentially possible to open and modify the same data by 2 users. This would result in overwriting and loss of data. To prevent it, the application features pessimistic lock mechanism. It preemptively locks access by other users to data modified by a co-worker.

Reported data lock is carried out in tables. Opening the table, even without data modification, locks the table exclusively. Other users can open a locked table, however, but it will be accessible in a read-only mode. Closing the table by the user who edited the table causes lock release. Tables are only visual presentations of data, therefore part of data modified by one user can be accessible through modifications in a different table. Such tables are locked in the sharing mode. A table is open to modifications if:

- it is not locked in any mode and
- none of the related tables is locked in the exclusivity mode and
- none of table related report data locks is active (e.g. someone else is editing report context settings)

`LOCK_TYPE` in the **`DBLOCK`** table has the following value

- **`TableExclusiveLock`** – locking the table for exclusive use,
- **`TableSharedLock`** – locking the table for the shared use,

Another case of the locking process relates to context modifications. Context changes influence the range of reporting data, and therefore also the range and visual shape of forms and tables. That is why all forms relating to the period under modifications are locked while modifying context. In case a form is open for modifications, context change is not possible. **`LOCK_TYPE`** has the following value

- **PeriodConfigurationLock**.

Apart from that, lock function is used to disable certain simultaneous administration, configuration and editing operations. For other users only a specific function is locked in such a case, e.g. if user tries to import data at given moment, the information message about the lock pops up. **LOCK_TYPE** can have the following values:

- **ReportDefLock** - reporting period management,
- **ReportDefReportCreationLock** - creating new reporting period,
- **ModuleNewReportCreationLock** - creating new report in given reporting period,
- **ModuleUserManagementLock** - upon entering the user management window,
- **ReportRoundingLock** - activating round report data function,
- **ReportGenerationLock** - generating XBRL instance,
- **EntityTaxonomySetLock** - rules management - edit, import, etc.
- **ReportRemovingDataLock** - report data removal function,
- **ReportCalculatingLock** - calculating report rules,
- **ReportImportingSharedLock** - importing report data (in shared mode),
- **ReportConsolidationLock** - generation of report data calculated in the process of other reports data aggregation,
- **ReportStatusModificationLock** - changing report status,
- **ReportArchivieCreationLock** - creating a new report archive,
- **ReportArchiveManagementLock** - report archives management,
- **TableSavingLock** - saving report table data,
- **ModuleJmsMigrationLock** - importing JMS messages from external source (file),
- **ReferencesLock** - editing report table cell additional descriptions-comments,
- **LoginBlockingLock** - locking work in aSISt with an external process (e.g. data migration),
- **LoginLock** - logging into the system,
- **ModuleProfilerDbSpeedLock** - start test measuring speed of database operations,
- **MappingSetRO** - presentation of the details for last fetching by mappings for selected report cell

4.2.1 Locks in external (independent) modules

- **EntityRoleLock** - editing role (Users and Roles module),
- **EntityExclusionLock** - editing exclusion (Corrections module),
- **EntityCorrectionItemLock** - editing correction (Corrections module),
- **EntityConsolidationDefLock** - exclusions and corrections registry management (Corrections module),
- **EntityConsolidationDefLock_RO** - exclusions and corrections registry management in read-only mode (Corrections module),
- **KeyStoreLock** - keys and systems access data management (Key Store module),
- **JDBCDriverDefLock** - database drivers management (Key Store module),

- **MappingSetCreate** - creating mapping set (Mapping module),
- **MappingDs** - mappings data sources management (Mapping module),
- **MappingDsCreate** - creating mappings data source (Mapping module),
- **MappingSetW** - mapping set management (Mapping module),
- **EntityDataRepositoryDefLock** - editing the definition of a data source (Data Repository module),
- **EntityCatalogueVariableLock** - editing the catalogue variable (Catalogue Variables module),
- **EntityAnalyticalReportLock** - editing the analytical report (Analytical Reports module),
- **EntityProspectusLock** - editing brochure (Brochures module),

4.2.2 Locks in tranSIS System

- **ReportTransisImportingFileLock** - importing report data from file (tranSIS System),
- **EntityTransisMasterContextLock** - editing context (settings) of reporting period (tranSIS System),
- **EntityTransisMessageActionLock** - processing JMS message (tranSIS System),
- **ModuleJmsReportImportLock** - importing report data from JMS message (tranSIS System),
- **ModuleBanksManagementLock** - bank units list management (tranSIS System)

5 Data feeding interface

5.1 General information

aSISt provides a set of functions which enable data feeding. The user may choose a way of data import (data are overwritten, records including already saved data will be ignored, data will be removed prior to import etc.). Additionally, the user can choose from 3 different data import formats. Following sections describe data import in a format compliant with XBRL standard, CSV and XML files.

5.2 XBRL data import

Importing from XBRL format allows loading reported data from XBRL instances format.

Validation is based on entry data compliance check with the obligatory taxonomy for a given reporting period.

Due to information saved in the XBRL instance, and concerning the reporting period for which the report is prepared, it is not possible to load an XBRL file generated for another reporting period.

5.3 CSV data import

5.3.1 Import description

Data from CSV is imported to specific aSISt documents (XBRL templates).

Entry data validation against the target document is performed by checking measure and dimension codes entered into the import file. Additionally, in case of data structures list (tuples), an imported value can be indexed, and in case of values reported for different periods – identified with a time stamp.

Data imported in CSV format need to have a linear structure, separated with a new line symbol CR/LF (two characters with ASCII 13 and 10) codes.

The imported file needs to be in UTF8 format, without the UTF mark (of hexadecimal representation EFBBBF) at the beginning of the file.

5.3.2 Data set structure – description

First row has to include column headers (their definitions are provided in the description of data format):

taxonomy;measure;dimension;index;period;value

Value of the **taxonomy** column is an enumeration which defines the report type (CC – Corep EBA, CF – FINREP EBA, CLE – Large exposure EBA, etc).

Single data line has the following format (semicolon serves as the field separator):

<tax code>;<measure code>;<dimensions code>;<structure index>;<time stamp>;<value>

where:

<taxcode> enumeration value, which represents report type.

<measure code> is an explicit designation of the value measure.

In case a measure is included in structures (tuple), the code is built on the basis of an identifier, structure measure and identifier, or component measure, separated with a colon:

<identifier/structure label>:<identifier/component label>

In case the component is a structure, the process of combining identifiers or measures is performed recursively:

<identifier/structure label>:<identifier/component structure label>:<identifier/component label> …

Measure and dimension code roles may be performed by XBRL identifiers or technical labels. It is possible to use one type of labels in a single import file – code type is selected in aSISt user interface upon the import.

<dimensions code> is an explicit designation of the combination of dimensions and its components. In case of values reported in a multi-dimensional context, individual codes are separated with a comma:

<dimension code>,<dimension code>,<dimension code> …

<dimension code> comprises of an identifier or a technical label of the dimension and the dimension component tag, separated with a colon:

<identifier/dimension label>:<dimension component tag>

<dimension component tag> is an XBRL identifier or a technical label for predefined dimensions. For user-defined (typed) dimensions, the identifier is created on the basis of the dimension component value, according to the following algorithm:

for components comprising of multiple fields, it is combined into one string, separated by the underscore
in the received string white characters and semicolons, commas, colons, backslashes, less-than signs, more-than signs, quotation marks and apostrophes are replaced with underscores

<dimensions code> can be an empty value.

<structure index> is an ordinal number of list elements. Only tuples can serve as list elements. All facts which belong to one structure copy must have the same index. Indexes for the same structures are natural numbers, beginning with **1**.

In case of embedded structures, structure index is as follows:

<number>:<number>

Where: the first number is the superior structure index, and the second one – substructure index.

Indexes are used in forms with lists, e.g.:

- FBN026 Information on related entities
- FBN031 Subordinated loans
- FIN025 Market risk
- FIN026 Interbank deposits
- FIN027 Involvement per country
- FID Information on bank

<period tag> defines a moment in time for which a given fact is reported. This tag needs to be specified only for facts reported for different periods, in the remaining case it can be defined automatically. It can have one of 3 following values:

S – period start

E – period end

D – duration

The period itself depends on aSISt configuration.

<value> is the reported value. If a value contains a semicolon, new line or quotations marks, it must be closed with quotation marks. Additionally, quotation marks in value content must be doubled.

In case of numbers, a comma serves as a decimal separator.

A set **<measure code>**, **<dimensions code><structure index><period tag>** must be unique in an imported file.

5.3.3 Entry data format in BNF notation

<file> ::= <header><line> { <line> }

<header> ::= ["template" ";"] "measure" ";" "dimension" ";" "index" ";" "period" ";" "value""\n"

<line> ::= [<form code>";"]<measure code>";" [<dimensions code>] ";" [<structure index>] ";" [<period tag>] ";" <value>"\n"

<measure code> ::= (<measure identifier> { ":"<measure identifier> })

| (<measure label> { ":"<measure label> })

<dimensions code> ::= <dimension code> { "," <dimension code> }

<dimension code> ::= (<dimensions identifier> | <dimension label>)

":" <dimension component tag>

<dimension component tag> ::= <dimension component identifier>

| <dimension component label> | <defined component identifier>

<structure index> ::= <number> { ":" <number> }

<period tag> ::= ("S" | "E" | "D")

<value> ::= ["""] (<text> | <number>) ["""]

<form code>, **<measure identifier>**, **<dimension identifier>**, **<dimension component identifier>**, **<measure label>**, **<measure label>**, **<measure component label>** are terms of XBRL taxonomy.

<defined component identifier> is built on the basis of the above presented algorithm.

<number> is a natural number.

<text> is a text string, in which potential quotation marks are doubled

<number> is a number with a full stop and a decimal separator, without a thousands separator.

5.3.4 Examples

```
measure;dimension;index;period;value
FXX00092;FDPP000:FDPP004,FDTZ000:FDTZ004,FDWA000:FDWA025;;;24324
FXX00092;FDPP000:FDPP008,FDTZ000:FDTZ004,FDWA000:FDWA002;;;48930
FIN05007;FDTZ000:FDTZ002,FDWA000:FDWA025;;;57843
FXX00092;FDPP000:FDPP004,FDTZ000:FDTZ007,FDWA000:FDWA025;;;69854
```

Example 1

```
measure;dimension;index;period;value
FID00075:FID00072;;1;;Jan Tadeusz
FID00023;;;;54-440
```

Example 2

```
measure;dimension;index;period;value
FIN25002:FIN25007;;1;;453665,56
FIN25002:FIN25007;;2;;432554,64
FIN25002:FIN25006;;1;;432545,30
FIN25002:FIN25006;;2;;456654,54
FIN25002:FIN25006;;3;;356340,68
```

Example 3

```
measure;dimension;index;period;value
FXX00072;;;S;694354
FXX00072;;;E;698594
FBN24007;;;D;90342
```

Example 4

5.4 XML data import

5.4.1 Import description

Data from XML is imported to specific aSISt documents (XBRL templates).

Entry data validation against the target document is performed by checking measure, dimension and period tag codes entered to the import file.

5.4.2 Data set structure – description

A file with imported data should be a well-formed XML document. Additionally, the first line contains an XML declaration with coding designation.

Example declaration:

```
<?xmlversion="1.0"encoding="UTF-8"?>
```

5.4.2.1 <report> element

<report> element, which contains all imported data, is the root of an XML file. This element has three attributes:

- **labelType** – an attribute defining the type of identifiers used in XML document. Only two values are permitted:
  - **Technical –** technical identifiers. Measure and dimension code roles are be performed by technical labels.
  - **Normal –** XBRL identifiers. Measure and dimension code roles are be performed by XBRL identifiers.

  labelType – a required attribute.

- **name** – The value of this attribute is not interpreted during data import. The attribute is present due to the requirement of compliance with aSISt versions 1.X.

  name – an optional attribute.

- **type** – enables defining a taxonomy for imported data. The following values are permitted:
  - **COREP –** for data imported to forms from COREP taxonomy.
  - **FINREP –** for data imported to forms from FINREP taxonomy.

This attribute can be ignored, its lack does not introduce any ambiguities (they only occur when the fact ID is unique within all taxonomies).

type – an optional attribute.

The value of <report> element is a sequence of <item> or<tuple> elements, where <item> elements represent single facts or component structures, and <tuple> elements represent structures.

5.4.2.2 <item> element

<item> element represents a single fact in a specific aSISt document (form). This element has following attributes:

- **periodType** – attribute defines a moment in time for which a given fact is reported. It can have one of 3 following values:
  - **S** – period start
  - **E** – period end
  - **D** – duration

  periodType - is a required attribute.

- **measureId** - is an explicit designation of a measure. Depending on the label type (labelType attribute) it is a technical label or an XBRL identifier.

  In case of a measure included in structures (tuple), the code is built on the basis of an identifier or a component measure:

measureId = "<identifier/component label>"

measureId – a required attribute.

- **dimensionId** – is an explicit designation of a combination of dimensions and its components. In case of values reported in a multi-dimensional context, individual codes are separated with a comma.

  dimensionId = "<dimension ID>,<dimension ID>,<dimension ID>…"

  <dimension ID> comprises of an XBRL identifier or a technical label (depending on the value of labelType attribute of the report element) of the dimension and dimension component tag, separated with a colon:

  <identifier/dimension label>:<dimension component label>

  <dimension component label> for predefined dimensions is an XBRL identifier or a technical label of a component, depending on the value of labelType attribute, report element.

  For user-defined (typed) dimensions, the identifier is created on the basis of the dimension component value, according to the following algorithm:

  for components comprising of multiple fields, it is combined into one string, separated by the underscore in the received string white characters and semicolons, commas, colons, backslashes, less-than signs, more-than signs, quotation marks and apostrophes are replaced with underscores

  dimensionId - can be an empty value.

  dimensionId - is a required attribute.

A reported fact value is a value of **<item>** element. In case of numbers, a comma serves as a decimal separator.

5.4.2.3 <tuple> element

<tuple> element represents structures (tuples). This element has one attribute which defines the structure measure:

- **measureId** – is an explicit designation of a structure measure. Depending on the code type (labelType attribute) it is a technical label or an XBRL identifier.

  measureId = "<identifier/structure label>"

  For tuples included in lists, their order in an imported file implicates the order on the list.

  Forms containing a list, e.g.:

  - FBN026 Information on related entities
  - FBN031 Subordinated loans
  - FIN025 Market risk
  - FIN026 Interbank deposits
  - FIN027 Involvement per country
  - FID Information on bank

measureId - is a required attribute.

The values of **<tuple>** element can have a form of <tuple> elements (embedded structures) or single facts (<item>).

Example structure representation:

```
<tuple measureId="structure_id">
        <item measureId="structure_component_ID" dimensionId="dimension" periodType="E">
                value
        </item>
</tuple>
```

5.4.3 Entry data format described with XML Schema

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:simpleType name="T_PeriodType">
<xs:restriction base="xs:string">
<xs:enumeration value="E"/>
<xs:enumeration nvalue="S"/>
<xs:enumeration value="D"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="T_TaxonomyType">
<xs:restriction base="xs:string">
<xs:enumeration value="FINREP"/>
<xs:enumeration value="COREP"/>
</xs:restriction>
</xs:simpleType>
<xs:simpleType name="T_LabelType">
<xs:restriction base="xs:string">
<xs:enumeration value="Technical"/>
    <xs:enumeration value="Normal"/>
</xs:restriction>
</xs:simpleType>
<xs:complexType name="T_Tuple">
<xs:choice>
<xs:element ref="item"maxOccurs="unbounded"/>
<xs:element ref="tuple"maxOccurs="unbounded"/>
</xs:choice>
<xs:attribute ref="measureId"use="required"/>
</xs:complexType>
<xs:complexType name="T_Item">
<xs:simpleContent>
<xs:extension base="xs:string">
```

```
<xs:attribute ref="periodType"use="required"/>
<xs:attribute ref="measureId"use="required"/>
<xs:attribute ref="dimensionId"use="optional"/>
</xs:extension>
</xs:simpleContent>
</xs:complexType>
<xs:complexType name="T_Report">
<xs:sequence>
<xs:group ref="fact"maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute ref="type"use="optional"/>
<xs:attribute ref="name"use="optional"/>
<xs:attribute ref="labelType"use="required"/>
</xs:complexType>
<xs:attribute name="type"type="T_TaxonomyType"/>
<xs:attribute name="periodType"type="T_PeriodType"/>
<xs:attribute name="name"type="xs:string"/>
<xs:attribute name="dimensionId"type="xs:string"/>
<xs:attribute name="measureId"type="xs:string"/>
<xs:attribute name="labelType"type="T_LabelType"/>
<xs:element name="report"type="T_Report"/>
<xs:element name="item"type="T_Item"/>
<xs:element name="tuple"type="T_Tuple"/>
<xs:group name="fact">
<xs:choice>
<xs:element ref="item"/>
<xs:element ref="tuple"/>
</xs:choice>
</xs:group>
</xs:schema>
```

5.4.4 Sample data

```
<?xml version="1.0" encoding="UTF-8"?>
<report type="FINREP" labelType="Technical">
<item measureId="DAN003"dimensionId="" periodType="E">i_test</item>
<item measureId="DAN004"dimensionId="" periodType="E">n_test</item>
<item measureId="DAN005"dimensionId="" periodType="E">s_test</item>
<item measureId="DAN006"dimensionId="" periodType="E">t_test</item>
<item measureId="DAN007"dimensionId="" periodType="E">e_test</item>
</report>
```

Example 1 (technical identifiers)

```
<?xml version="1.0"encoding="UTF-8"?>
<report type="FINREP" labelType="Technical">
<tuple measureId="FIN25002">
<item measureId="FIN25003"dimensionId=""periodType="E">2007-08-17</item>
<item measureId="FIN25005"dimensionId=""periodType="E">1,00</item>
<item measureId="FIN25006"dimensionId=""periodType="E">2,00</item>
<item measureId="FIN25007"dimensionId=""periodType="E">3,00</item>
<item measureId="FIN25008"dimensionId=""periodType="E">4,00</item>
<item measureId="FIN25009"dimensionId=""periodType="E">5,00</item>
<item measureId="FIN25010"dimensionId=""periodType="E">6,00</item>
<item measureId="FIN25011"dimensionId=""periodType="E">7,00</item>
<item measureId="FIN25012"dimensionId=""periodType="E">8,00</item>
<item measureId="FIN25013"dimensionId=""periodType="E">9,00</item>
</tuple>
</report>
```

Example 2 (technical identifiers)

```
<?xml version="1.0"encoding="UTF-8"?>

<report type="FINREP"labelType="Technical">
<item measureId="FBN01004"dimensionId="FDWA000:FDWA025"periodType="E">2,00</item>
<item measureId="FBN01005"dimensionId="FDWA000:FDWA025"periodType="E">4,00</item>
<item measureId="FBN01004"dimensionId="FDWA000:FDWA002"periodType="E">1,00</item>
<item measureId="FBN01005"dimensionId="FDWA000:FDWA002"periodType="E">3,00</item>
</report>
```

Example 3 (technical identifiers)

```
<?xml version="1.0"encoding="UTF-8"?>

<report type="COREP"labelType="Technical">
<item measureId="CXX0005"dimensionId="CDTY007:typeDim"periodType="E">1,00</item>
<item measureId="CXX0002"dimensionId="CDTY007:typeDim"periodType="E">2,00</item>
</report>
```

Example 4 (technical identifiers)

```
<?xml version="1.0"encoding="UTF-8"?>
<report type="FINREP"labelType="Normal">
<item measureId="p-data_CreatorFirstName"dimensionId=""periodType="E">i_test</item>
<item measureId="p-data_CreatorLastName"dimensionId=""periodType="E">n_test</item>
<item measureId="p-data_CreatorLastName"dimensionId=""periodType="E">n_test</item>
<item measureId="p-data_CreatorPhoneNumber"dimensionId=""periodType="E">t_test</item>
<item measureId="p-dane_CreatorEmailAddress"dimensionId=""periodType="E">e_test</item>
```

```
</report>
```

Example 5 (XBRL identifiers)

```
<?xml version="1.0"encoding="UTF-8"?>
<repor ttype="FINREP"labelType="Normal">
<tuple measureId="p-FINREP-pl_MarketRiskDailyOpenCurrencyPositions">
<item easureId="p-FINREP-pl_Data"dimensionId=""periodType="E">2007-08-17</item>
<item measureId="p-FINREP-pl_OpenPositionsUSD"dimensionId=""periodType="E">1,00</item>
<item measureId="p-FINREP-pl_OpenPositionsEUR"dimensionId=""periodType="E">2,00</item>
<item measureId="p-FINREP-pl_OpenPositionsCHF"dimensionId=""periodType="E">3,00</item>
<item measureId="p-FINREP-pl_OpenPositionsGBP"dimensionId=""periodType="E">4,00</item>
<item measureId="p-FINREP-pl_OpenPositionsJPY"dimensionId=""periodType="E">5,00</item>
<item measureId="p-FINREP-
pl_OpenPositionsRemainingExchangeableCurrencies"dimensionId=""periodType="E">6,00</item>
<item measureId="p-FINREP-pl_OpenPositionsExchangeableCurrencies"dimensionId=""periodType="E">7,00</item>
<item measureId="p-FINREP-pl_OverallPosition"dimensionId=""periodType="E">8,00</item>
<item measureId="p-FINREP-pl_CapitalRequirement"dimensionId=""periodType="E">9,00</item>
</tuple>
</report>
```

Example 6 (XBRL identifiers)

```
<?xml version="1.0"encoding="UTF-8"?>

<report type="FINREP"labelType="Normal">
<item measureId="p-FINREP-pl_CashInTill"dimensionId="d-FINREP-pl-cu_CurrenciesDimension:d-FINREP-pl-
cu_OtherThanPln"periodType="E">2,00</item>
<item measureId="p-FINREP-pl_RemainingFundsInTill"dimensionId="d-FINREP-pl-cu_CurrenciesDimension:d-FINREP-pl-
cu_OtherThanPln"periodType="E">4,00</item>
<item measureId="p-FINREP-pl_CashInTill"dimensionId="d-FINREP-pl-cu_CurrenciesDimension:d-FINREP-pl-
cu_Pln"periodType="E">1,00</item>
<item measureId="p-FINREP-pl_pl_RemainingFundsInTill"dimensionId="d-FINREP-pl-cu_CurrenciesDimension:d-FINREP-pl-
cu_Pln"periodType="E">3,00</item>
</report>
```

Example 7 (XBRL identifiers)

```
<?xml version="1.0"encoding="UTF-8"?>

<report type="COREP"labelType="Normal">
<item measureId="p-cm-ca_CreditRiskCapitalRequirements"dimensionId="d-
ty_ConsolidatedSubgroupsRegulatedEntitiesDimension:typeDim"periodType="E">1,00</item>
<item measureId="p-cm-ca_SettlementRiskCapitalRequirements"dimensionId="d-
ty_ConsolidatedSubgroupsRegulatedEntitiesDimension:typeDim"periodType="E">2,00</item>
```

```
</report>
```

Example 8 (XBRL identifiers)

5.5 PD data import

5.5.1 Import description

Data import in PD format is possible only for reports representing a daily balance. PD format was prepared by the supervisor (in this case: the National Bank of Poland). Data generated from aSISt or tranSIS to the supervisory entity are delivered in this format. To facilitate data feed into a PD report, aSISt enables data import in this format.

5.5.2 Data set structure – description

PD format set is a fixed-point file with a following structure:

Field	Field length	Decimal places	Contents
1	4 or 8 characters	---	Number of the bank which prepares the report (in case the number has less than 4 characters, it needs to be filled with lead spaces (on the left side)).
2	10	---	Report date – YYYY-MM-DD format E.g. 1985-11-08
3	5	---	Report identifier, e.g. ID001 (ZZZZZ for any control record)
4	2	---	Row ID, from the row with a given information, e.g. A1 (ZZ for any control record).
5	2	---	Column ID, for the column with a given information (A1 – PLN, B1 – Foreign currency) (ZZ for a control record on a day level, ZI for a record with a number of records in a set, ZN – with a report version number within the reporting period)
6	15	2	Value conforming to the column on a form, specified with a row and column ID (for aggregate positions – total for a specific day, number of rows with sums in a month or number of report versions for a given month).

Row and column IDs are construed on the basis of last measure and dimension characters respectively. Value of the dimension defined by the user is created on the basis of date value from field 2.

Additionally, an imported file, apart from records representing reported amounts, has special records which enable entering control sums.

5.5.3 Example PD file

```
Line 1: 12342010-03-01PD001A2A1    5555555.00
Line 2: 12342010-03-01PD001A2B1      66666.00
Line 3: 12342010-03-01PD001B2A1    7777777.00
```

```
Line 4: 12342010-03-01PD001B2B1     88888.00
Line 5: 12342010-03-01PD001C2A1   9999999.00
Line 6: 12342010-03-01PD001C2B1     10101.00
Line 7: 12342010-03-01PD001D2A1     15562.00
Line 8: 12342010-03-01PD001E2A1       201.00
Line 9: 12342010-03-01PD003L1A1       112.00
```

5.6 F1 data import

5.6.1 Import description

Data import in F1 format is possible only for F1 forms. F1 format was created by the National Bank of Poland and is used to send reports to a supervisory entity. To facilitate data feeding, aSISt enables data import in this format.

5.6.2 Data set structure – description

A file in F1 format consists of two record groups: header data and data.

Header data include the following fields:

- FRM – form (possible value: ‘fa’ or ‘fb’)
- BANK – 4-digit bank ID
- PRD – period for which the report is prepared (in 3-MM-YYYY format)
- TIME – time of report generation (in DD-MM-YYYY HH:MM[:SS] format)
- PRC – name of the report creator
- PERS.AUTH. – name of the authorized person

Value of header data is represented in a single line conforming to the following pattern:

```
;<header_data_type><value>
```

Reported values are transferred to record groups in accordance with the following pattern:

```
<row_identifier>;<value>;
```

Row ID is construed on the basis of data order presented in aSISt (1 is and ID of the first row of data, 2 of the second one, etc.).

5.6.3 Sample file in F1 format

```
;FRM fa
;BANK 0000
;PRD 3-03-2010
```

```
;TIME 30-04-2010 10:30:19
;PRC Jan Kowalski
;PERS.AUTH. Katarzyna Nowak
1;1234;
2;5678;
3;910;
4;;
5;;
6;;
7;;
8;;
9;;
10;1000;
```

5.7 F7 data import

5.7.1 Import description

Data import in F7 format is possible only for F7 reports. F7 format was created by the National Bank of Poland and is used to send reports to a supervisory entity. To facilitate data feeding, aSISt enables data import in this format.

5.7.2 Data set structure – description

A file in F7 format consists of two record groups: header data and data.

Header data include the following fields:

- FRM – form (possible value: 'IP')
- BANK – 4-digit bank ID
- PRD – period for which the report is prepared (in 3-MM-YYYY format)
- TIME – time of report generation (in DD-MM-YYYY HH:MM[:SS] format)
- PRC – name of the report creator
- PERS.AUTH. – name of the authorized person

Value of header data is represented in a single line conforming to the following pattern:

```
;<header_data_type><value>
```

Reported values are transferred to record groups in accordance with the following pattern:

```
<row_identifier>
;<1_column_value>;<2_column_value>;<3_columne_value>;<4_column_value>;<5_column_value>;<6_column_value>
```

5.7.3 Sample file in F7 format

```
;FRM f7
;BANK 0000
;PRD 3-03-2010
;TIME 30-04-2010 10:30:19
;PRC Jan Kowalski
;PERS.AUTH. Kasia Nowak
1;31;45;46;23;11;45;
2;24;65;34;65;23;43;
3;43;36;76;65;22;63;
4;57;24;76976;37;42;
5;66;67;90;76;23;54;
6;66;57;87;69;26;63;
7;82;42;78;56;95;91;
8;76;67;34;67;14;16;
```

5.8 PEGAZ data import

Data import in PEGAZ format is possible only for PEGAZ reports. This format is used by the National Bank of Poland to send reports from aSISt and tranSIS. In order to facilitate data feeding into aSISt, the user can import data in the format in which they are sent to the supervisor.

5.8.1 Data set structure – description

Data in PEGAZ format are XML documents conforming to the schemes defined by the National Bank of Poland in XML Schema standard.

5.9 Excel data import

5.9.1 Import description

The application supports two systems of data import from Excel files. Data from Excel files is imported to specific aSISt documents (XBRL templates). The first system of Excel files loading is based on absolute offsets calculated on the basis of header tables' size (rows and column). In later versions of aSISt this import mechanism might be removed.

The second system is based on calculations of a cell address, on the basis of row and column header labels. That is why the header order does not need to conform to what is presented in aSISt as well as in a single Excel table. The user may define a connection between multiple separate tables from aSISt. One requirement is a valid structure of row and column headers for each cell with data.

The application automatically decides on the data loading system from an Excel file. If a file does not contain a comment in A1 cell, the application chooses a system based on table headers. If a file contains a comment in A1 cell and does not contain a valid header, it will be loaded with an absolute offsets system.

5.9.1.1 Absolute offsets system – description

Entry data validation against the target document is performed by checking data entered to the import file. The file with imported data should be a valid Excel spreadsheet, where data to be imported are located on the first sheet. A1 cell should contain a comment in the following format **<table_identifier>;<transposition>**, where:

- **<table_identifier>** – is a system table ID. ID can be specified by exporting a selected table to Excel and reading the A1 cell comment.
- **<transposition>** – information on whether the table was transposed. Possible values: **'DEFAULT'** table was not transposed; **'TRANSPOSED'** table was transposed.

5.9.1.2 Example comments

FIN018:p-FINREP-pl_IRRPositionsPerRevaluationPeriodLengthPresentation:t-FINREP-pl-FIN018_FIN018Cube;DEFAULT
FIN018:p-FINREP-pl_IRRPositionsPerRevaluationPeriodLengthPresentation:t-FINREP-pl-FIN018_FIN018Cube;TRANSPOSED

5.9.1.3 Header-based system – description

Header-based system does not require a comment with table ID or table location specified (contrary to the absolute offsets-based import). An exception to this rule can be a table with an external size (an external size includes all table cells but is not a part of column or row headers). In such case, the user may add appropriate cells to the row or column header so that each cell contains an appropriate size or, for simplification, in A1 Excel cell add table information as a commentary (table ID contains default information on external size).

Due to the cell content analysis, it is also necessary to define labels used in row and column headers. aSISt, upon data import from Excel, supports the following header types:

- **MIXED_LABEL** – technical labels with additional descriptive label for a given concept. When loading a row and column headers, application analyzes only a part of the label – a technical label.

  " **[**" **<TECHNICAL_LABEL>** "**]**" **<DESCRIPTIVE_LABEL>**

- **ID_LABEL** – a technical label.
- **XBRL_ID_LABEL** – an XML concept ID.

Row and column headers in an import file should be compliant with the chosen label mode. In case of a header-enclosed value which cannot be found in a taxonomy, application will consider this header as a user's extension and will ignore it while creating individual cells. Row and column headers for data representing user's dimension should include, apart from the dimension value, information on ID dimension.

For **MIXED_LABEL** labels, the header representing user's dimension should be as follows:

- "["**<DIMENSION_TECHNICAL_LABEL>**":"**<DIMENSION_VALUE>**"]" **<DESCRIPTIVE_LABEL>**

For **ID_LABEL** labels:

- **<DIMENSION_TECHNICAL_VALUE>**":"**<DIMENSION_VALUE>**

And for XBRL_ID_LABEL labels:

- **<DIMENSION_XBRL_ID>**":"**<DIMENSION_VALUE>**

An exceptional case of concept values in row and column headers are tables containing the same fact with different periods (one at the beginning of the period, the other one at the end). In this case, it is necessary to provide additional information which allows to differentiate fact values from the beginning and the end during data loading process. To indicate a fact from the period beginning, it is essential do add a header of postfix##S measure.

BNF notation for A1 cell comment can be presented in the following way:

[<table_identifier>[";"<ANY>][";"<LABEL_MODE>][";"<ANY>]

where:

- **<table_identifier>** – was defined in the description of the absolute offsets-based import
- **<LABEL_MODE>** – possible values are: **MIXED_LABEL, ID_LABEL, XBRL_ID_LABEL**
- **<ANY>** – any parameter which will not be interpreted by the application

In case of an ignored comment in an Excel file with ignored information on label mode, the application will assume default labels in **MIXED_LABEL** mode.

6 SIS module

6.1 Basic information

"SIS" module introduces a new communication channel which enables sending and validation report statuses directly from the application.

To send or validate a current XBRL report status, SIS module communicates with NBP service developed in SOAP Web Service technology.

Until now, aggregate reports were sent via a web service in a form of single XBRL documents containing reporting data of individual banks. Now, they have been replaced with a set of independently identifiable, individual XBRL

reports of respective banks. Summarizing, the term "aggregate report", from the point of view of the NBP service, seized to exist.

For functional reasons (optimization and simplification of report group delivery, users' "habits") the term "aggregate report" still exists, however, its function is limited to logical grouping of reports sent in a single operation.

NBP SIS service defines two basic functionalities (implemented as synchronic WEB Service methods):

Send report to bank (including report validation with regards to its technical correctness)

Check current report validation status (identified through a unique ID received as the result of a correctly performed operation no. 1)

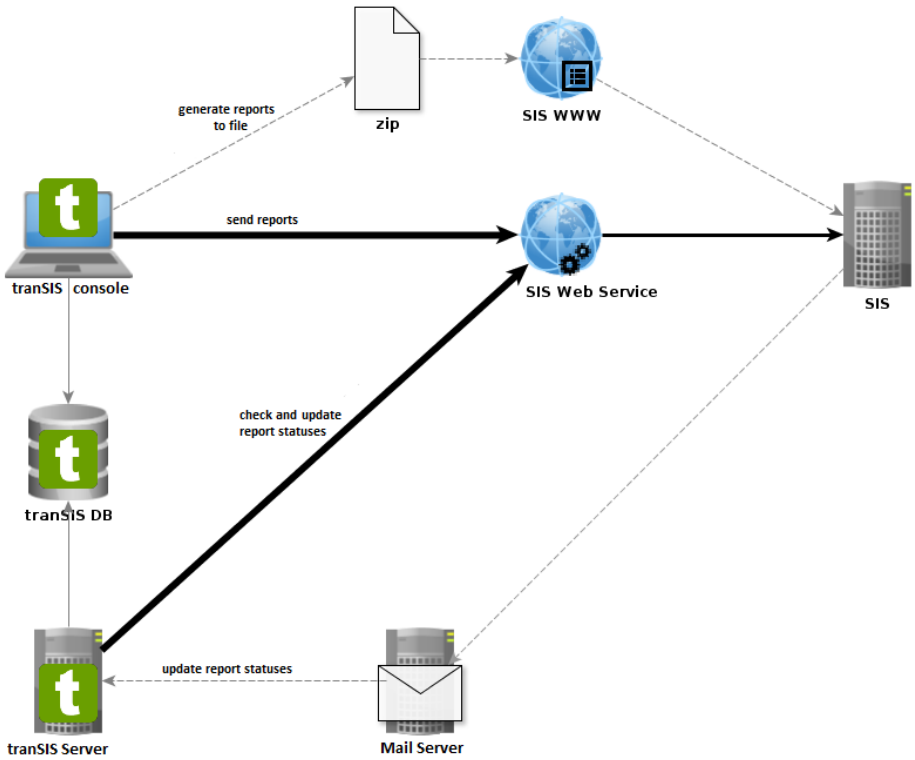


Figure 3. tranSIS - SIS communication

The default communication channel is "SIS Web Service", however, in one of the steps of the "aggregate" generation process, the user can choose "pack" generation of selected reports to a file, in order to hand them in "an emergency mode" to SIS via a web service.

In the default mode (communication via Web Service), the application presents the user online with results of the handing over process of successive reports which are a part of an "aggregate report".

Additionally, components of an "aggregate report", presented in the tranSIS console, were enlarged with the following information:

- communication channel chosen during generation process (Web Service, "File in SIS 1 format" or "File in SIS 2 format")
- current report processing status (of an aggregate report component) in SIS service ("during validation", "positions verified", "incorrect")
- information on errors in the report returned by SIS service during validation

The tranSIS Console user, from the application, can "at will" check in SIS the current validation status of a chosen report.

Setting up a new communication channel requires SSL certificates support, which are defined by NBP security policy and are necessary to authenticate the application in SIS Web Service. During run-up, the application imports to the session (from "certs" sub-catalog) a required certificate which enables communication with SIS service.

6.2 Technology

Communication between the work environment and the SIS Web Service was developed in "Apache CXF" technology. Support of certificates and communication through SSL was developed with standard "Oracle Java JDK" mechanism from "javax.net.ssl" pack.

6.3 Configuration

Access to functionalities relating to communication with NBS SIS is possible only after SIS module activation in the user license.

6.3.1 Certificates

Certificates necessary for authentication in NBP SIS service must be copied to "certs" sub-catalog (e.g. "tranSIS/certs/").

6.3.2 NBP SIS URL configuration

URL of NBP SIS web service is defined by the "**sis.webservice.URL=**" line (e.g. "sis.webservice.URL = http://localhost:9998/services/") in an update file of the application:

- aSISt (db.properties)
- tranSIS (transis.properties)

7 aSISt – remote program update

aSISt allows for remote application updates. The default URL for downloading updates is https://support.asist-xbrl.eu/updates/update_config.xml However, updating application in the Internet is not always possible due to restrictions imposed by the security policy and for many other reasons. Remote application update can be configured in a local network, which improves the update process.

7.1 Remote update configuration

7.1.1 Configuration on website server – description

Updates are described with an XML document which indicates a list of available updates and a URL address of the catalogue in which the files with updates are located. Remote update configuration is based on a properly prepared configuration file and saved update files. An example update configuration file is as follows:

```
<updateConfig>
<baseURL>http://host.com/catalog/catalog1/</baseURL>
<updatesList>
<updateDescr>
<url>update_name_1.5.jar</url>
<formVersion>1.4</formVersion>
<toVersion>1.5</toVersion>
</updateDescr>
<updateDescr>
<url>update_name_1.6.jar</url>
<formVersion>1.5</formVersion>
<toVersion>1.6</toVersion>
</updateDescr>
</updatesList>
</updateConfig>
```

<updateConfig> element is the root of an XML file. This element aggregates <baseUrl> and <updatesList>.

<baseURL> element specifies a catalogue in which updates are located. In this example all updates are downloaded from:

http://host.com/catalog/catalog1/<update_file_name>

Another element aggregated in <updateConfig> is <updateList>. This element contains an update list. The order of update descriptions is not set. Elements required while defining an update are:

<url> – the value of this element describes the name of the update file. Concatenation of <baseURL> and <url> element value indicates a location of the update file.

<formVersion> – value of this element indicates the aSISt version for which the update is intended.

<toVersion> – value of this element indicate application the version for which it will be updated.

Two updates are described in the example below:

- First update:

```
<updateDescr>
<url>update_name_1.5.jar</url>
<formVersion>1.4</formVersion>
<toVersion>1.5</toVersion>
</updateDescr>
```

This update is suitable for aSISt v1.4 and aSISt will be updated to v1.5. The update file can be found under the following address:

http://host.com/catalog/catalog1/update_name_1.5.jar

- Second update:

```
<updateDescr>
<url>update_name_1.6.jar</url>
<formVersion>1.5</formVersion>
<toVersion>1.6</toVersion>
</updateDescr>
```

This update is suitable for aSISt v1.5 and aSISt will be updated to v1.6. The update file can be found under the following address:

http://host.com/catalog/catalog1/update_name_1.6.jar

In most cases, local network administrators can download the published configuration file from the following address **http://asist.net.pl/updates/update_config.xml** as well as all updates described therein, and then change the base update URL (<baseURL> element) to a URL indicating a catalogue, available in the local network, containing downloaded update files.

8 XBRL Generation

8.1 Filing Indicator

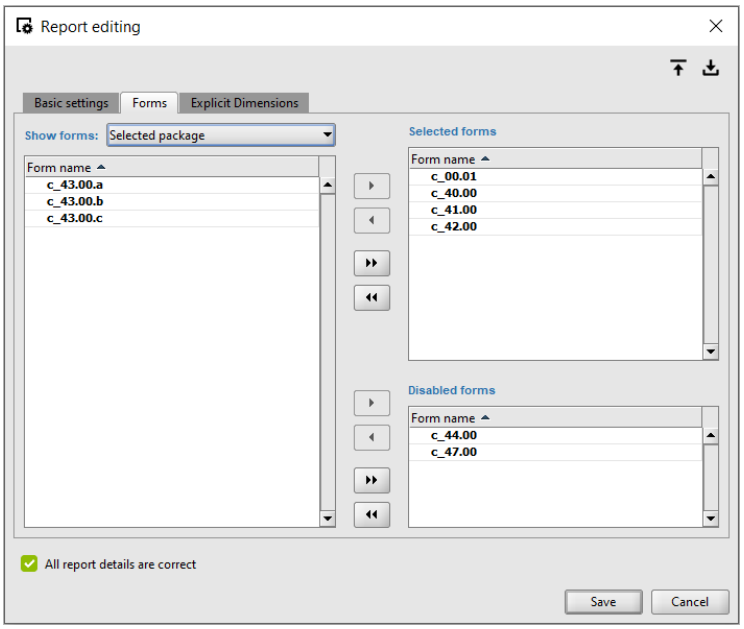
In the EBA and Solvency taxonomies the supervisor defines the Filing Indicator. The Filing Indicators are used to express information, which form was reported or was not reported. In aSISt application manage of the Filing Indicators is automatic and based on following rules:

- "true" – the status is defined for all forms, which are available in the taxonomy and are chosen in the report context, regardless of the data filling. "True" status refers to empty and filled forms.
- "false" – the status is defined for all forms, which are available in the taxonomy and are not chosen in the report context.

"Disabled forms" from the report will not be included in the generated file.

There is not any connection between filing indicators and the table which is filled in by the user as a declaration of completion of the forms. The filing indicator is independent of the value declared in that table in a technical way.

The status of the filing indicator is generated in XBRL file.

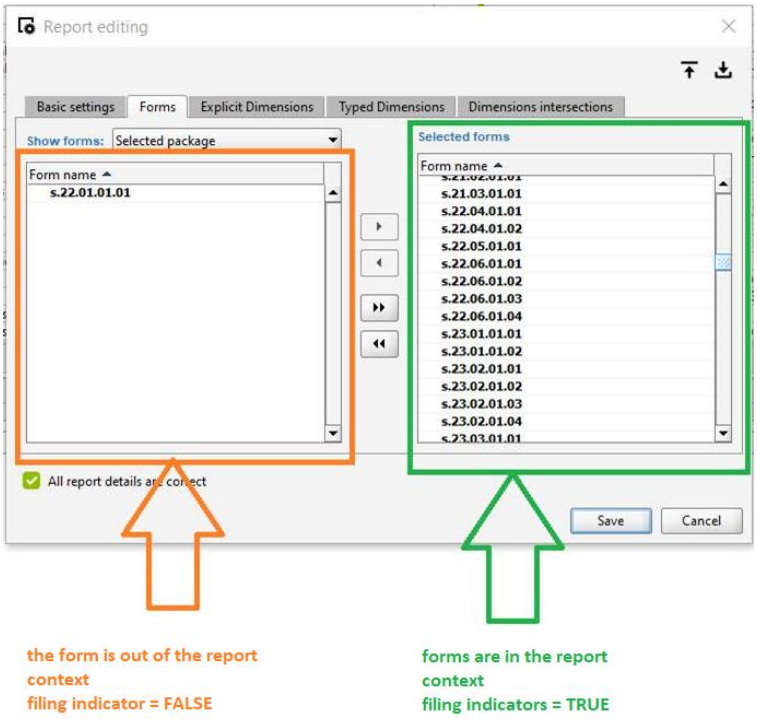


In the case of reports which forms are divided into many sub-items, it is enough that at least one of them is in the "Selected forms" section, then in the generated report, the Filing Indicator with the status "true" will be assigned to the form.

8.1.1 Example for SOLVENCY II taxonomy

Filing indicator is not related in any way to the information contained in form s.01.01.

Form s.01.01 is filled in by the user and it is only an indication of data reporting status on individual forms. The filing indicator is independent of the value declared in the s.01.01 in a technical way.
The filing indicator presents which form is in the report context and which form is out of the report context – as in the picture below:



The notification generated to the XBRL file as below:

<find:filingIndicator id="ft_2138002WE79XZSYJB750_f762" find:filed="false" contextRef="c_17e46e">S.22.01</find:filingIndicator

means, that form s.22.01 is not in the report context.