# Description of XML Import

XML Import is done in the following steps:

1. Loading of XML files.
2. The first parsing of XML data.
3. Matching of the objects and references with the target configuration:
    - matching of the imported objects with the target configuration
    - ID is assigned to new objects (insert) if the ID reservation failed or ID was zero
    - matching of the object references with the target configuration
    - checking the consistency of the parents of the changed objects (update), while the change is allowed:
        - for the object of I/O tag, Event, Database, Table, Eval Tag, Alarm, Remote Tag, Historical Value types
        - for other types of objects (e.g. Line, Station) provided that the parent process (e.g. KOM) is stopped. If the parent of the object is directly a process (e.g. for objects of type Line), both the old and the new parent process must be stopped
          Note: check of parent process running in redundant systems is performed only on the HOT D2000 Server. If the parent process is running on the SBS D2000 Kernel, it will be stopped after the XML import is finished (and if Autostart is configured, it will be started again).
    - matching of the column references with the target configuration
4. Reading of the binary files.
5. The second parsing of XML data.
6. XML data import.

## XML data parsing

Each **XML file** together with the proper binary files represents the object configuration that is to be imported.

The conditions of successful parsing:

- XML file structure must be correct
- each object reference <HOBJ_REF> in <REFERENCES> section must have:
    - filled name <name>
    - unique name <name> and <uid> (if it is filled) within this section
- each column reference <COL_REF> in <REFERENCES> section must have:
    - filled name <col_name>
    - unique name <col_name> and <col_idx> (if it is filled) within this section
    - <col_idx> must not acquire a value higher than MAX_VALID_COL_IDX = 7777
- the structure <tObjItemData> must be listed as the first in <CFGRECORDS> section
    - name <Name> of the structure must be filled
    - if the object in references refers to itself, its name must match with uid

If an object or column reference is found while parsing <CFGRECORDS> section (it was not found according to name in prepared references from <REFERENCES> section), it will be added automatically into <REFERENCES> section. It means that the references from <REFERENCES> section are just a sort of some rule that is primary taken into consideration but the import may be done also without <REFERENCES> section.
This action is followed by a check whether all the objects (as individual elements) were changed from XML to the structures in memory. It only involves the check of objects (whether they are correct or not) regardless of their membership in the particular configuration.

## Matching of the imported objects with target configuration

The principle of object matching is shown on the flowchart of object matching.

An important element at matching is **uuid** in **TObjItemData** structure.
The second important element is **Name** in **TObjItemData** structure.

Warnings that may occur at matching:

- the object is being updated via NAME after the entered UID of object was not found (see 4. and 5. example in the flowchart)
- the object is being updated and the value type of object is different from the value type of object in target configuration
- the object is being inserted and the entered ID of object was not reserved in target configuration

Errors that may occur at matching:

- value of UID or NAME is the same as value of another imported object
- object intended for import relates to object that just has been deleted from target configuration
- loads such the UID from the target configuration that already exists among the imported objects (see 2. and 4. example in picture)
- object is being renamed but this NAME has another object in the target configuration (see 7. example in picture)
- import of new object with name that already belongs to other object in the target configuration (see 6. example in picture)
- object intended for update is of different type than object in the target configuration
- parent object of the imported object is not set
- parent object of the object intended for update is different than the parent object of the imported object (except the object I/O tag, Event, Database or Table because they allow the changes)

## Matching of the object references with target configuration

The principle of object references matching is shown on the flowchart of object references matching.

The important element at matching is **uid** in **HOBJ_REF** structure.
The second important element is **name** in **HOBJ_REF** structure.
If the parameter IGNR_REFS is checked, the matching may be executed only through the second element.

Warnings that may occur at matching:

- name, type, or value type of object is different in object reference than in the object which refers to
- object of reference is searched via NAME after the reference could not be connected via UID (only if import parameter UID_STRICT = OFF)

Errors that may occur at matching:

- the object which the object reference refers to can not be found

## Matching of the column references with target configuration

The principle of column references matching is shown on the flowchart of the column references matching.

The important element at matching is **col_idx** in **COL_REF** structure.
The second important element is **col_name** in **COL_REF** structure.
If the parameter IGNR_REFS is checked, the matching may be executed only through the second element.

Warnings that may occur at matching:

- name or value type of column is different in column reference than in the column that refers to

Errors that may occur at matching:

- the column, to which the column reference refers, can not be found

**Related pages:**

D2000 XML
XML Import
Object matching - flowchart