

4.8.2 CAMT: Assignment logic

 Language: [Deutsch](#) · [English](#)

4.8.2 CAMT: Assignment logic

Assignment logic – part of the CAMT Nuclet documentation (import and processing of CAMT.053/CAMT.054 bank statements (Cash Management)).

HOW-TO

INTEGRATORS

UPDATED: JUL 2026

APPLIES TO NUCLOS 4.2026.X

On this page

- [Related pages](#)



Machine-translated – under review

This page was machine-translated from German as a first draft and is currently under review. The authoritative source remains the **German original** (see the language switch above).

The assignment of incoming bank transactions to reference objects (invoices, etc.) is implemented in the `findMatchingReferences()` method of the abstract class `org.nuclet.camt.logic.AbstractCAMT054Logic`. By default, the assignment is based on the extracted reference number of the reference objects:

- If the reference number is found in the remittance information of a bank transaction, an assignment is made.
- Otherwise no assignment is made.

This logic is expressed by the expression `boBankTransaction.getInformationToAccountOwner().indexOf(strReference) >= 0` (from the code segment, see below).

If this rule is to be modified (e.g. so that the check is less strict), this is the place to start. If possible, the change should not be made directly in the `AbstractCAMT054Logic` class. The source-code location intended for this in the concrete implementation `CAMT054Logic` is marked accordingly with an `@replace!` tag.



It is recommended to make the changes in the concrete implementation `CAMT054Logic` or in your own class that inherits from the abstract class `AbstractCAMT054Logic`.

```

/**
 * Matches the given bank transaction with a map of references:
 *
 * If the transaction's field "information to account owner" contains the reference, the related
 * BusinessObject will be memorised, i.e. a list of all matching entries will be returned.
 *
 * @note A more sophisticated, application specific, behaviour might be implemented in
 * dedicated subclasses
 *
 * @param boBankTransaction a BusinessObject of type <code>BankTransaction</code>
 * @param mpReferences a map of references to be matched
 * @return a list of all BusinessObject that relate to a matching reference
 *
 * @throws BusinessException might be thrown by implementing classes in case of errors or other exceptions
 */
protected List<T> findMatchingReferences(final BankTransaction boBankTransaction,
    final Map<T, String> mpReferences)
throws
    BusinessException
{
    final ArrayList<T> lstMatchingReferences = new ArrayList<T>();
    final Set<T> stReferencedBusinessObjects = mpReferences.keySet();

    // Check, if a matching reference exists...
    for (final T businessObject : stReferencedBusinessObjects) {
        final String strReference = mpReferences.get(businessObject);

        logTrace("checking \"" + strReference + "\"...");

        if (strReference != null || strReference.trim().length() == 0) {
            if (boBankTransaction.getRemittanceInformation().indexOf(strReference) >= 0) {
                log("matching reference found: " + strReference);

                lstMatchingReferences.add(businessObject);
            }
        } else {
            logWarn("");
        }
    }
    return lstMatchingReferences;
}

```

Related pages

Nuclet: CAMT (available in the trade Nuclet)

CAMT Nuclet data sheet

[Open](#)

Interfaces

All integrations

[Open](#)