Microsoft Dynamics® AX 2012

Using the Enhanced Number Sequence Framework in Microsoft Dynamics AX 2012

White Paper

This document presents an overview of changes made to the number sequence framework in Microsoft Dynamics AX 2012. It includes typical ISV scenarios and partner customization scenarios for the setup and use of number sequences in the enhanced number sequence framework.

Date: August 2011

Author: Madan Natu, Senior Program Manager

Send suggestions and comments about this document to adocs@microsoft.com. Please include the title with your feedback.

Table of Contents

Introduction

Number sequencing is an essential feature in an Enterprise Resource Planning (ERP) system. Microsoft Dynamics® AX employs a number sequence framework to generate alphanumeric number sequences. These sequences can be used for transaction documents such as sales orders, purchase orders, invoices, and journals; or for master data entities such as customers, vendors, and employees. The primary purpose of the number sequence framework is to provide unique, user-friendly identifiers while maintaining a continuous or noncontinuous alphanumeric sequence.

Number sequences in earlier versions

The number sequence framework in Microsoft Dynamics AX 2009 and earlier versions supports only hard-coded or literal suffixes and prefixes, and a sequential system-generated alphanumeric string. The user is allowed to specify hard-coded literal strings for suffixes and prefixes for a number sequence format. This is done in the context of setting up a number sequence when the user is also specifying values for the smallest and largest numbers in the sequence, number sequence code, and name. The framework allows a generated alphanumeric string to be split into multiple parts by hard-coded strings, but this capability is only required in very rare scenarios.

The Microsoft Dynamics AX 2009 number sequence framework also supports features such as number sequence groups, sharing of a specific number sequence for multiple documents, and an API using a number sequence code or ID (for example, a voucher series corresponding to journal names). These patterns are still supported in the Microsoft Dynamics AX 2012 number sequence framework, although they will not be discussed in this white paper.

Reasons for enhancing the number sequence framework

The changes that have been made to the number sequence framework were motivated primarily by a paradigm shift that occurred with the introduction of the new organization model in Microsoft Dynamics AX 2012. A second motivation was to generalize the framework to support extensibility in future releases. The Microsoft Dynamics AX 2009 framework relies on a DataArea, or company-based, infrastructure and is limited to defining number sequences for a specific company. With the new organizational model, it is possible to define global entities that are not tied to any specific company or even to a legal entity or operating unit. For global master data entities such as product or employee, this requires the ability to generate number sequences that apply globally.

The new organizational model in Microsoft Dynamics AX 2012 provides for both a legal entity structure and an operating unit structure for an organization. Transaction or master data entities can now be related to operating unit or legal entity, rather than just to a company. Some countries or regions, such as France and China, have financial and accounting regulatory requirements for generating external or internal transactions such as invoices, sales orders, and journal vouchers that are based on a date field. A legal entity that is based in France, for example—but is also part of a multinational corporation—is required to generate documents that have a unique and continuous number sequence within a given fiscal calendar period. This means that, to satisfy regulatory requirements, they must include a prefix/suffix of the fiscal calendar period as part of the number sequence to identify the document as pertaining to that period. They must also include another prefix/suffix, due to organizational requirements, to identify the document as originating in the French legal entity.

Audience

This white paper is primarily targeted at ISV and partner developers who want to perform customizations when using the number sequence framework. However, administrators may be interested in the "Setup and administration of number sequences" section, which describes how to configure and set up number sequences using the new number sequence administration forms.

Overview

Microsoft Dynamics AX 2012 introduces two concepts to the number sequence framework: *segment* and *scope*. A *segment*, which is synonymous with *parameter*, is a data entity such as legal entity, operating unit, or fiscal calendar period that can be used to define a number sequence. In the enhanced number sequence framework, a number sequence can have more than one segment. A *scope* is a valid combination of segments used for a specific transaction or master data entity. By using segment and scope, Microsoft Dynamics AX 2012 extends the capabilities of the number sequence framework.

All segments in a scope must be related to the underlying transaction or master data entity. For example, a journal transaction will have a relationship with the segments of legal entity and fiscal calendar period (through the transaction or posting date) that make up its scope. The scope can be used to partition the entity instances or transactions based on the segment values. In this example, a journal transaction might be partitioned into categories based on the two segments in its scope.

A unique number sequence code must be created for every possible combination of legal entity and fiscal calendar period instances. For example, if there are two legal entities with IDs 10 and 20, and there are fiscal calendar periods for every calendar month in 2011, the user must define separate number sequences for each combination. An example of a scope instance is 20-Jan11. An example of a number sequence code for a journal might be **JN-J-20-Jan11**, in which JN represents a journal. The number sequence format, **J-20-Jan11,** is a string that is defined by the end user to represent the identifier for transactions for legal entity 20 and fiscal calendar period January 2011. The actual instance of a journal might have a generated number sequence such as J-20-Jan11-000340. (The last number in the sequence, "000340", is a system-generated number.)

The concept of a number sequence *reference*, which is synonymous with an *extended data type*, has been carried forward from Microsoft Dynamics AX 2009. When you create a number sequence, you must first create an extended data type (EDT) with a name —such as *SalesID*—for the new number sequence. The label of this EDT is used as a reference. This reference is used to define a field on a document or master data entity that requires a number sequence.

Supported scopes

We are converting a subset of existing transactions and master data entities to allow them to make use of scopes such as legal entity or shared scope. This enables Microsoft Dynamics AX 2012 to ship with predefined scopes for this subset of entities. The rest of the data entities will still have a default scope of Company (or DataArea).

Microsoft Dynamics AX 2012 does not currently support an arbitrary combination of segments as a valid scope. The following are the only scopes that are currently valid in the number sequence framework:

- Shared
- Company (DataArea)
- Company (DataArea) and fiscal calendar period
- Legal entity
- Operating unit
- Legal entity and fiscal calendar period

In a shared scope, a generated number will be shared across an instance of Microsoft Dynamics AX 2012, and has no segments. An example of shared scope in Microsoft Dynamics AX 2012 is the number generated for a Purchase Requisition document.

Although scope is predefined for a data entity, the segments within that scope are configurable. That is, if a scope consists of legal entity and fiscal calendar period, a user can decide to configure the scope to be only legal entity during implementation. There are, however, some mandatory restrictions on scope composition. If, for example, fiscal calendar period is selected as part of a scope, the inclusion of DataArea or legal entity is mandatory in that scope.

One limitation of the new number sequence framework is that the scope itself must always be configured for a specific reference. You cannot define a scope based on variations in the data. For example, suppose the regulatory requirements in France and China require that you generate a number sequence by using the legal entity and fiscal calendar period segments. Other countries or regions, however, might not have the same requirements. The administrator in a multinational corporation might decide that it is not necessary to include both segments for all countries or regions. However, you cannot define two different scopes—one to be used in France and China that includes legal entity and fiscal calendar period segments, and another to be used in all other countries or regions that includes only the legal entity segment. You could use the framework in such a way that all countries or regions have the scope of legal entity and fiscal calendar period, but display the number sequence format based on both segments only in France and China. However, the numbers generated would still be partitioned based on both segments in all companies.

A second limitation is that we are not shipping any references with an out-of-the-box fiscal calendar period segment. Extending the framework to include a fiscal calendar period currently requires customization of existing parameters forms in specific modules. The new **Number sequences** setup form does support a fiscal calendar segment.

Setup and administration of number sequences

When an end user creates a document such as a sales order, purchase order, or invoice, the extended data type on the document field—such as SalesID or InvoiceID—is passed to the number sequence framework. In addition, information is passed for the segments such as DataArea, or legal entity plus fiscal calendar period, depending on the scope for the reference. The framework then generates a number sequence code based on these parameters (segments), and this number sequence code is used to generate a number based on the format for the corresponding number sequence.

Specifying a number sequence is now done in two steps: segment configuration and setup. In Microsoft Dynamics AX 2009, segment configuration was not required because the implicit default segment was Company (or DataArea), and the setup of all number sequences was done in the context of that single segment. The Microsoft Dynamics AX 2012 number sequences framework allows multiple segments, but legacy number sequence references do not support this new capacity. Customers and partners who plan to perform customizations to enable new segments, based on the new capabilities of the framework, will need to use the **Segment configuration** form. This form can be used to view or modify the default configuration of segments shipped out of the box.

Segment configuration

An administrator can configure the segments that are allowed for their requirements. For example, out of the box, the application might allow two segments such as legal entity and fiscal calendar period. However, only the legal entity segment might be included by default. The administrator can enable both segments for their scenario by using the **Segment configuration** form.

In Microsoft Dynamics AX 2012, most references use either Company (or DataArea) or legal entity segments out of the box. There are several references for master data entities that are shared across the application and do not use segments. However, the framework allows an administrator to add segments such as fiscal calendar period. Figure 1 displays the **Segment configuration** form, which can be accessed by using the navigation path **Organization Administration** > **Common** > **Number sequences** > **Segment configuration**.

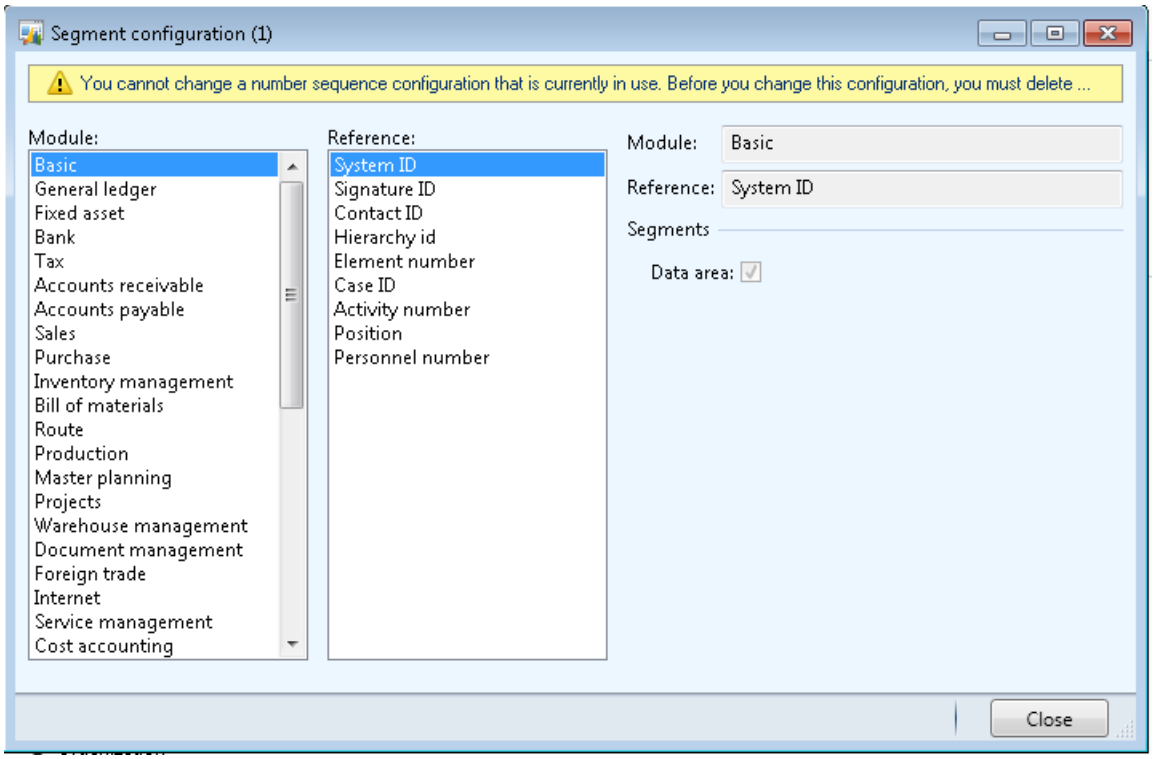

Figure 1: Segment configuration form

Setup

The existing number sequences can be viewed by using the **Number sequences** list page. The navigation path to this page is **Organization administration** > **Common** > **Number sequences** > **Number sequences**. Figure 2 shows the **Number sequences** list page.

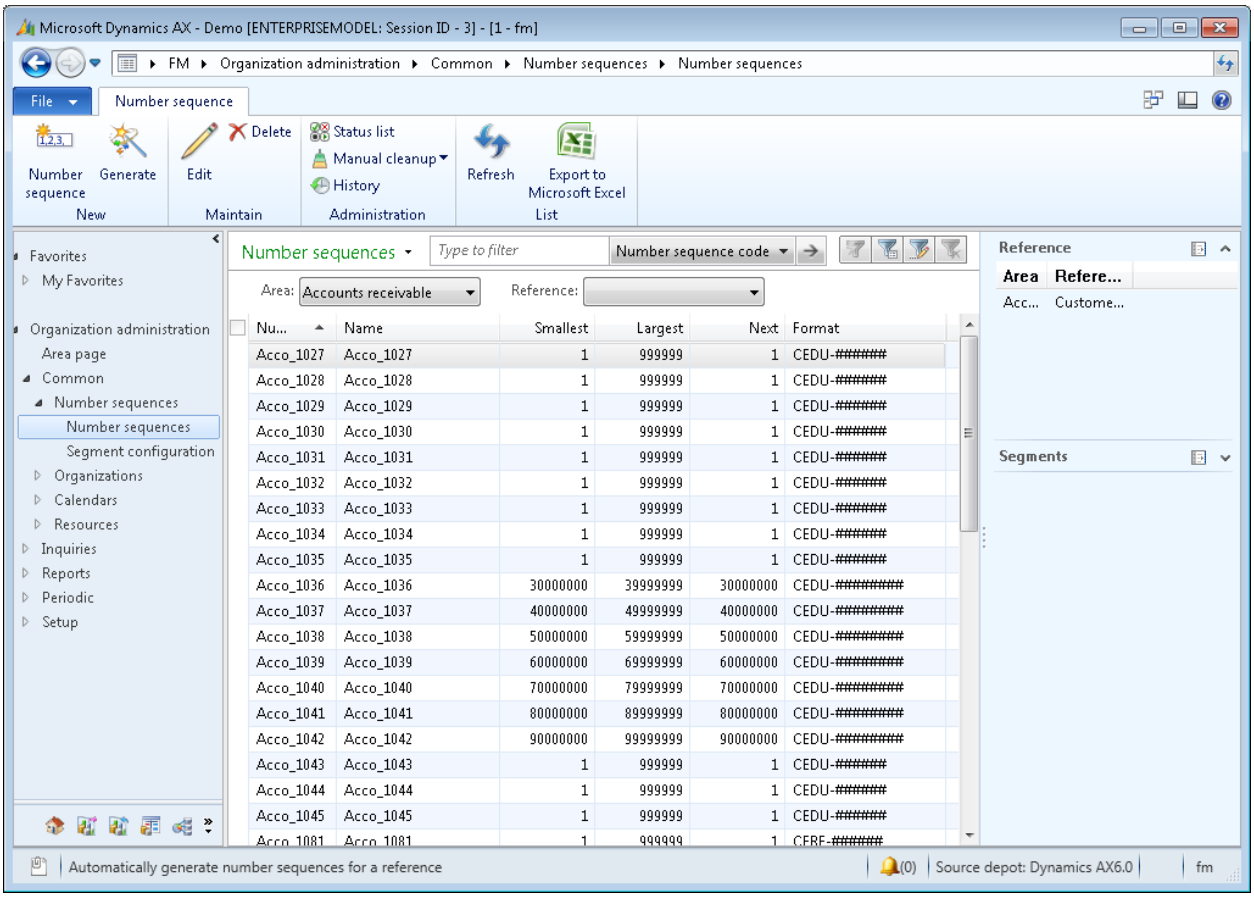

Figure 2: Number sequences list page

All of the actions that were available in the Microsoft Dynamics AX 2009 forms are still available in the action pane of the list page. In addition, the user can now run the **Number sequence wizard** by clicking **Generate** in the **New** group of the action pane. The wizard will create number sequences for all references that do not have number sequences defined in the system.

In Microsoft Dynamics AX 2009, the wizard was run separately for each company to generate the number sequences within each company. In Microsoft Dynamics AX 2012, the new enhancements to the number sequence framework support segments other than company. Therefore running the wizard is independent of the selected company in the workspace.

The wizard uses the short name fields on the tables underlying the segments to define a format. You can use the actions of the wizard ("Include scope in format" and "Remove scope from format") to include or remove segment values within a format, as needed. The number sequence code and the smallest and the largest values can be changed. However, you cannot select custom filters on the list page; the wizard cannot generate number sequences by selected areas or references. Figure 3 shows the Number sequence wizard.

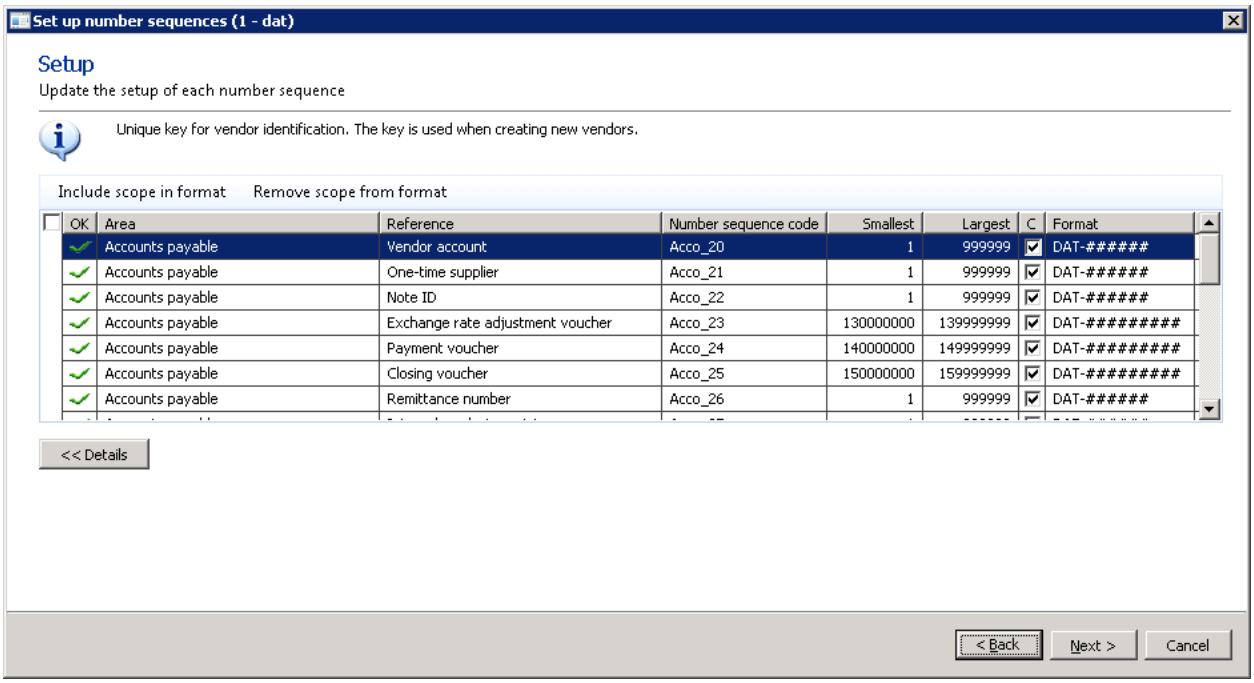

Figure 3: Number sequence wizard

On the Microsoft Dynamics AX 2012 **Number sequences** list page shown in Figure 2, the user can select an area —for example, **Accounts receivable**—and a specific reference, such as **Customer account**, to see all number sequences defined for all customer accounts across all organizations. The user can also open a specific number sequence by double-clicking a selected number sequence. This opens the **Details** page as shown in the following illustration.

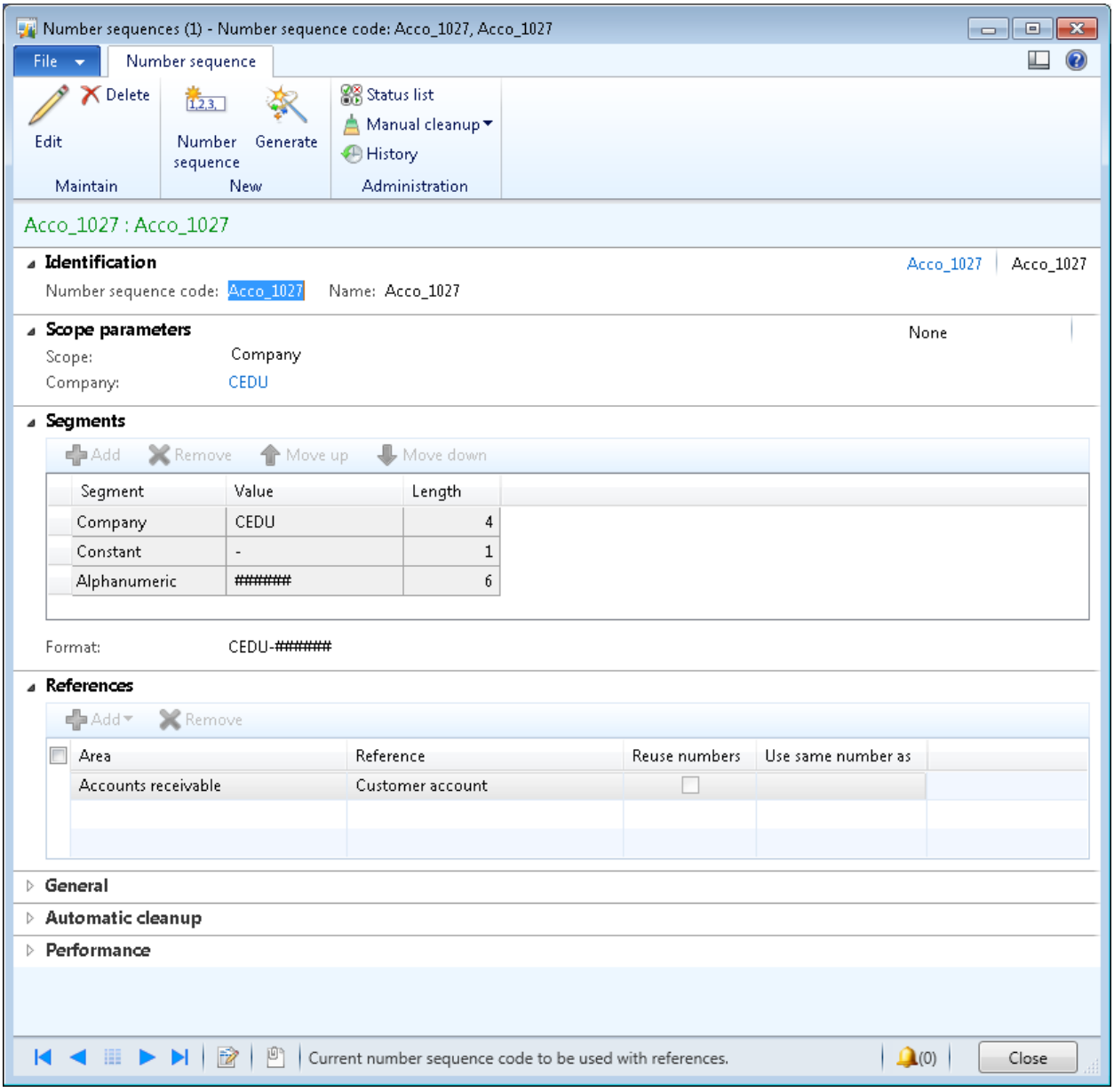

Figure 4: Number sequence details

To create a new number sequence:

1. On the **Number sequences list page**, in the **New** group, click **Number sequence**. The details page opens.
2. Enter a new code and name for the new number sequence on the **Identification** FastTab.
3. Select a scope on the **Scope parameters** FastTab. Depending on the scope selected, enter the appropriate segment values. For example, if the scope is Company, select the company for which this number sequence is being defined.

   Note Although the Legal entity and Company scopes are technically equivalent, for ease of use, we have decided to keep them as separate scopes.

   - The Company scope is used for all references for which the underlying table is a "per company" table that uses the DataAreaId field and whose SaveDataPerCompany property is set to "Yes". For example, this scope is used for the number sequence for the customer account number in the Customer table (CustTable).
   - The Legal entity scope should be used for references for which the underlying table is not a "per company" table and which holds a foreign key (FK) to the CompanyInfo (also known as "legal entity") table. The number sequence for the expense report number in the Expense Report (TrvExpTable) table uses this scope.
4. Define the format for the new number sequence by adding new segments and by using predefined segment values on the **Segments** FastTab. Rearrange the segments according to your required format. Literals are entered as constant segments, and actual numbers are specified as alphanumeric segments. Microsoft Dynamics AX allows you to enter an alphanumeric segment by using the wildcard symbol "#" for numbers and "?" for letters.

   The format for the segments is generated as an alphanumeric number sequence. Microsoft Dynamics AX also allows you to split this number sequence into multiple parts, consisting of constant and alphanumeric segments. At least one alphanumeric segment must be present. Segments such as company or legal entity are not mandatory in the format definition. However, they are still used for partitioning the numbers generated for a reference, based on the selected scope.
5. Add a new reference on the **References** FastTab to assign this number sequence to a reference. This step is not mandatory for sequences defined for special application usage patterns that return a new number by passing in the value of a number sequence code or ID without using a reference.

   Note The enhanced number sequence framework supports patterns that were previously supported in the Microsoft Dynamics AX 2009 API, which return a new number by passing in the value of a number sequence code without using a reference (for example, in a voucher series used for specific journal names). However, use of these patterns is not recommended.
6. Specify whether the number sequence is manual, and continuous or noncontinuous on the **General** FastTab. Also enter the lowest and highest number in the number sequence series in this FastTab. Enter other relevant information for this number sequence here.
7. Save the number sequence and close the form.

Administration

The administration of number sequences is performed by using actions provided in the **Administration** group on the action pane on the **Number sequences** list page.

- **Status list**: Provides a list of numbers that have been generated for continuous number sequences, but which have not been committed to the database. The numbers are either currently being used in a user session, are reserved for future use in a user session, or are free for use if a new client user session requests a new number for a particular number sequence in the list. If a new number does not exist for a specific continuous number sequence, it is generated by the sequence number framework from the next value for that number sequence in the **Number sequence** table (**NumberSequenceTable**).
- **Manual cleanup**: Allows the administrator to manually clean up numbers in the status list. Use of this option is only recommended after an unexpected system failure; in such rare circumstances, numbers might not be automatically cleaned up.
- **History**: Provides the history of changes to the number sequences themselves.

A number of administrator tasks can be performed from the **Details** page. An administrator can, for example, schedule an automated periodic cleanup for every number sequence by entering intervals on the **Automatic cleanup** FastTab.

An administrator can also assign number sequences by using a page in the parameters forms in individual application modules. For example, you can view or assign the number sequences to specific references in the General ledger module. You can navigate to the form by using the path **General ledger** > **Setup** > **Parameters**.

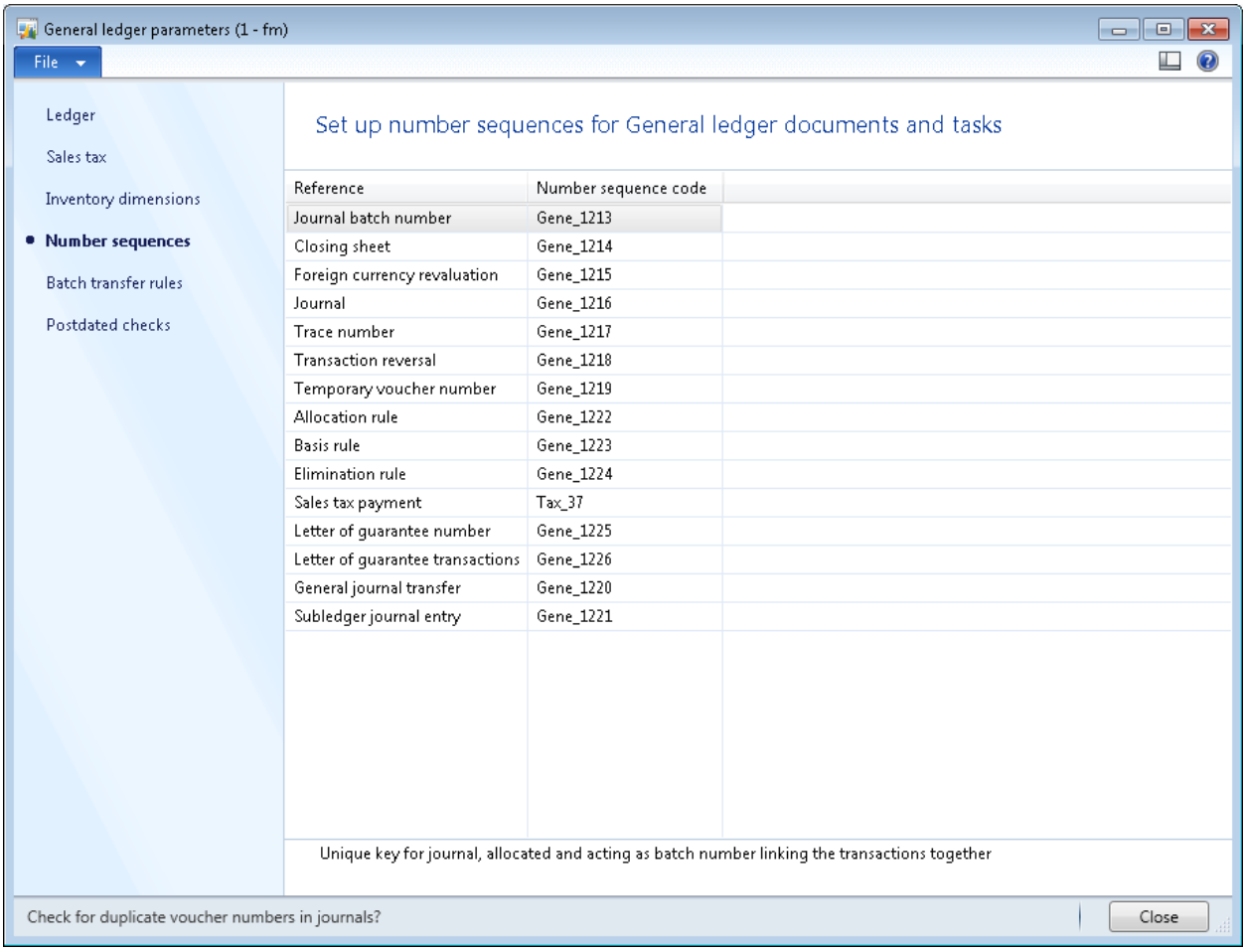

Figure 5: Number sequences for General Ledger

Performance considerations in number sequences

Performance must be taken into account when using the **continuous** flag on number sequences. A continuous number sequence only guarantees a gapless number sequence, not a truly continuous number sequence. The unused numbers (which are created but not used due to operations cancelled by the end user) are reused later if the **continuous** flag is checked. The use of continuous number sequences impacts system response times. Number sequence generation for transaction processing can be significantly improved if the number sequences are noncontinuous.

We recommend that you use noncontinuous number sequences unless there is a regulatory requirement for continuous number sequence, which is common for external documents such as purchase orders, sales orders, and invoices. Some numbers might be wasted and there might be gaps within the numbers, but the performance benefit is significant.

Also, for noncontinuous number sequences, you can set the preallocation quantity cache in the Performance FastTab. This cache helps improve system performance.

Extensibility scenarios

This section is primarily intended for ISV developers who want to call an API that extends the `NumSeqApplicationModule` class to handle changes required by the enhanced number sequence framework. This API is called to create new fields on documents or master data entities that will use a number sequence. The customization scenarios by partners follow the same pattern.

In a common extensibility scenario, a developer creates new fields and makes use of the new number sequence framework to generate values for those fields. There are two main extensibility scenarios:

- **Company scenario**: Based on the default segment of Company (or DataArea).
- **Organization model and regulatory scenario**: Based on the use of new segments such as legal entity, operating unit, and fiscal calendar period.

The Microsoft Dynamics AX 2012 number sequence framework does not currently support an extensibility scenario for adding arbitrary segments such as a warehouse or a site as a segment in the definition of a number sequence for a reference. That kind of extensibility would require significant customization to the number sequence framework.

In all extensibility scenarios, the assumption is that the developer is trying to set up a new module, has already defined a field in a table, and is using the new number sequence framework to generate values for that field.

A developer would typically follow these steps:

1. Make changes to support the new extended data type corresponding to the new field in the table.
2. Call the API to generate the values by using the number sequence for that field.

An extensibility scenario in which the developer changes the scope of the reference from DataArea to global (or shared) scope is also discussed.

Company scenario

This section provides the steps to implement a company scenario. This sample code is based on the Fleet Management example in Microsoft Dynamics AX 2009. Similar sample code for creating number sequences was provided in the Microsoft Dynamics AX 2009 documentation.

To implement the company scenario:

1. Create a new enum value **FM** with label **Fleet Management** in base ENUM `NumberSeqModule`.
2. Create a new `NumberSeqModule`*`XXX`* class, such as `NumberSeqModuleFleetManagement`, which extends the `NumberSeqApplicationModule` class. The sample code for creating this class is as follows.

```
Class declaration

public class NumberSeqModuleFleetManagement extends
  NumberSeqApplicationModule
{

}

protected void loadModule()
{
    NumberSeqDatatype datatype = NumberSeqDatatype::construct();

    /* Vehicle Number */

    datatype.parmDatatypeId(extendedtypenum(VehicleNum));
    datatype.parmReferenceHelp("Unique key for Fleet Management
  vehicles");
    datatype.parmWizardIsContinuous(false);
    datatype.parmWizardIsManual(NoYes::No);
    datatype.parmWizardIsChangeDownAllowed(NoYes::No);
    datatype.parmWizardIsChangeUpAllowed(NoYes::No);
    datatype.parmWizardHighest(999999);

    datatype.addParameterType(NumberSeqParameterType::DataArea, true,
  false);
    this.create(datatype);

    /* Trip Number */

    datatype.parmDatatypeId(extendedtypenum(TripNum));
    datatype.parmReferenceHelp("Unique key for trips");
    datatype.parmWizardIsContinuous(false);
    datatype.parmWizardIsManual(NoYes::No);
    datatype.parmWizardIsChangeDownAllowed(NoYes::No);
```

```
        datatype.parmWizardIsChangeUpAllowed(NoYes::No);
        datatype.parmWizardHighest(999999);

        datatype.addParameterType(NumberSeqParameterType::DataArea, true,
    false);
        this.create(datatype);


}

public NumberSeqModule numberSeqModule()
{
    return NumberSeqModule::FM;
```

Use of the DataArea segment in Step 2 to describe the default segment for the extended data types used for both vehicle number and trip number.

Note In Microsoft Dynamics AX 2009, number sequence references could be initialized by restarting the Application Object Server (AOS). In Microsoft Dynamics AX 2012, the initialization of references to populate the NumberSequenceDatatype and NumberSequenceParameterType tables has moved to the initialization checklist. To initialize the newly created references, run a job that executes the **LoadModule** method.

You can also reinitialize all references by running a job that executes the **LoadAll** method in the **NumberSequenceModuleSetup** class. However, for reinitializing all references, you must ensure that there are no existing number sequences already defined in the system..

3. Create a **Number sequences** page in the parameters form of the new module.

   See existing forms such as **CustParameters** or **LedgerParameters** for examples of the implementation. Here is sample code for the `numberSeqPreInit` method on the form.

```
void numberSeqPreInit()
{
    numberSequenceModules = [NumberSeqModule::FleetManagement];
    numberSeqApplicationModule = new NumberSeqModuleFleetManagement();
    scope = NumberSeqScopeFactory::createDataAreaScope();
    NumberSeqApplicationModule::createReferencesMulti(numberSequenceModul
  es, scope);


tmpIdRef.setTmpData(NumberSequenceReference::configurationKeyTableMulti(n
  umberSequenceModules));
}
```

Note This form can only be used for references that have a scope of DataArea. The administration forms described in the "Setup and Administration of number sequences" section can be used for references that have any scope. These forms can be found in **Organization Administration > Common > Number Sequences**.

4. In the business logic, use the API to generate a number sequence by using number sequence code.

   Note This step requires number sequence code to be set up by an administrator or a power user during the implementation.

   - Add a method on the parameter table (**FleetManagementParameters**) for the new module.

```
Public server static NumberSequenceReference numRefVehicleNumber()
  {
// Optional step for DataArea scope, mandatory for other scopes
NumberSeqScopeFactory::CreateDataAreaScope(selectableDataArea _dataArea =
  curext());
 return NumberSeqReference::findReference(extendedtypenum
  (VehicleNumber));
    }
```

   - Add a table method `FMVehicle::setVehicleNumber` for creating a new vehicle number.

```
Void setVehicleNumber()
{
NumberSeq num
NumberSequenceReference numberSequenceReference;

numberSequenceReference =
  FleetManagementParameters::numRefVehicleNumber();
if (numberSequenceReference)
{
num = NumberSeq::newGetNum(numberSequenceReference);
this.setField(fieldnum(FMVehicle, VehicleNumber),num.num());
}
}
```

Forms example

If you want to expose the number sequence field on a form in the client, you will typically add code to the data source for the form or data set.

To use a number sequence for a form:

1. In the class declaration of the form that will be accessing data, add a variable declaration for the number sequence handler. The following example shows the variable definition for a number sequence handler.

```
public class FormRun extends ObjectRun
{
    NumberSeqFormHandler numberSeqFormHandler;
}
```

2. Add the **NumberSeqFormHandler** method to the form. The code in this method will create an instance of the number sequence form handler and return it. The following example shows the code that returns the number sequence form handler for the Vehicle form of the Fleet Management sample module.

```
NumberSeqFormHandler numberSeqFormHandler()
{
    if (!numberSeqFormHandler)
    {
        numberSeqFormHandler = NumberSeqFormHandler::newForm(
        FleetManagementParameters::numRefFMVehicleNumber().NumberSequenceId,
        element,
        FMVehicle_DS,
        fieldnum(FMVehicle, VehicleNumber));
    }
    return numberSeqFormHandler;
}
```

3. Add **create**, **delete**, and **write** methods to the data source of the table that contains the field for which the number sequence is being used. The following code examples show these methods that are added to the data source for the FMVehicle table to support the number sequence for the VehicleNumber field.

```
public void create(boolean _append = false)
{
    element.numberSeqFormHandler().formMethodDataSourceCreatePre();
    super(_append);
    element.numberSeqFormHandler().formMethodDataSourceCreate();
}

public void delete()
{
    element.numberSeqFormHandler().formMethodDataSourceDelete();
    super();
}

public void write()
{
    super();
    element.numberSeqFormHandler().formMethodDataSourceWrite();
}
```

Enterprise Portal example

If you want to expose the number sequence field on a form in Enterprise Portal, you will typically add code to the data source for the form or data set.

To use a number sequence for a form in Enterprise Portal:

1. In the class declaration of the data set that will be accessing data, add a variable declaration for the number sequence handler. The following example shows the variable definition for a number sequence handler.

```
public class DatSetRun extends ObjectRun
{
    NumberSeqFormHandler numberSeqFormHandler;
}
```

2. Add the **NumberSeqFormHandler** method to the data set. The code in this method will create an instance of the number sequence form handler and return it. The following example shows the code that returns the number sequence form handler for the data set of the Fleet Management sample module.

```
NumberSeqFormHandler numberSeqFormHandler()
{
    if (!numberSeqFormHandler)
    {
        numberSeqFormHandler = NumberSeqFormHandler::newForm(
        FleetManagementParameters::numRefFMVehicleNumber().NumberSequenceId,
        element,
        FMVehicle_DS,
        fieldnum(FMVehicle, VehicleNumber));
    }
    return numberSeqFormHandler;
}
```

3. Add **create**, **delete**, and **write** methods to the data source for the data set that contains the field for which the number sequence is being used. The following code examples show these methods that are added to the data source for the FMVehicle table to support the number sequence for the VehicleNumber field.

```
public void create(boolean _append = false)
{
    element.numberSeqFormHandler().formMethodDataSourceCreatePre();
    super(_append);
    element.numberSeqFormHandler().formMethodDataSourceCreate();
}

public void delete()
{
    element.numberSeqFormHandler().formMethodDataSourceDelete();
    super();
}

public void write()
{
    element.numberSeqFormHandler().formMethodDataSourceWrite();
    super();
}
```

Organization model and regulatory scenario

The organization model and regulatory scenario is similar to the Company scenario. However, a few changes are required to reflect the new segments based on data entities such as legal entity, operating unit, and fiscal calendar period.

To implement the organizational model and regulatory scenario:

1. Create a new enum value **FM** with label **Fleet Management** in base ENUM NumberSeqModule.
2. This step is nearly identical to step 2 in the Company scenario, but adds a line of code (bolded below) that assigns a segment of fiscal calendar period to the new extended data type, **TripNum**.

```
Class declaration


public class NumberSeqModuleFleetManagement extends
  NumberSeqApplicationModule
{


}


protected void loadModule()
{
    NumberSeqDatatype datatype = NumberSeqDatatype::construct();

    /* Vehicle Number */

    datatype.parmDatatypeId(extendedtypenum(VehicleNum));
    datatype.parmReferenceHelp("Unique key for Fleet Management
  vehicles");
    datatype.parmWizardIsContinuous(false);
    datatype.parmWizardIsManual(NoYes::No);
    datatype.parmWizardIsChangeDownAllowed(NoYes::No);
    datatype.parmWizardIsChangeUpAllowed(NoYes::No);
    datatype.parmWizardHighest(999999);

 datatype.addParameterType(NumberSeqParameterType::DataArea, true,
  false);
 this.create(datatype);

    /* Trip Number */

    datatype.parmDatatypeId(extendedtypenum(TripNum));
    datatype.parmReferenceHelp("Unique key for trips");
    datatype.parmWizardIsContinuous(false);
    datatype.parmWizardIsManual(NoYes::No);
    datatype.parmWizardIsChangeDownAllowed(NoYes::No);
    datatype.parmWizardIsChangeUpAllowed(NoYes::No);
    datatype.parmWizardHighest(999999);
```

```
    datatype.addParameterType(NumberSeqParameterType::DataArea, true,
    false);
    datatype.addParameterType
    (NumberSeqParameterType::FiscalCalendarPeriod, true, false);
    this.create(datatype);
}

public NumberSeqModule numberSeqModule()
{
    return NumberSeqModule::FM;
  }
```

3. Create a **Number sequences** page in the parameters form of the new module.

   See existing forms such as **CustParameters** or **LedgerParameters** for examples of the implementation. Here is sample code for the numberSeqPreInit method on the form.

```
void numberSeqPreInit()
{
    numberSequenceModules = [NumberSeqModule::FleetManagement];
    numberSeqApplicationModule = new NumberSeqModuleFleetManagement();
    scope = NumberSeqScopeFactory::createDataAreaScope();
    NumberSeqApplicationModule::createReferencesMulti(numberSequenceModul
  es, scope);

tmpIdRef.setTmpData(NumberSequenceReference::configurationKeyTableMulti(n
  umberSequenceModules));
}
```

   Note This form can only be used for references that have a scope of DataArea. The administration forms described in the "Setup and Administration of number sequences" section can be used for references that have any scope. These forms can be found in **Organization Administration > Common > Number Sequences.**

4. Instantiate the scope for the two segments of Company (or DataArea), and fiscal calendar period, and call the API with these two segments passed as parameters to the number sequence framework. Use the following sample code:

   - Add a method on the parameter table (**FleetManagementParameters**) for the new module.

```
Public server static NumberSequenceReference
numRefTripNumber(TransDate _date = systemdateget())
  {

NumberSeqScope scope =
  NumberSeqScopeFactory::CreateDataAreaFiscalCalendarPeriodScope(curext()
  ,
FiscalCalendars::findPeriodByPeriodCodeDate
(CompanyInfo::fiscalCalendarRecId(),_date).RecId);
```

```
return NumberSeqReference::findReference(extendedtypenum(TripNumber),
  scope);
    }
```

- Add a table method `FMTrip::setTripNumber` for creating a new trip number.

```
Void TripNumber()

{
NumberSeq num
NumberSequenceReference numberSequenceReference;

numberSequenceReference = FleetManagementParameters::numRefTripNumber();
If(numberSequenceReference)
{
num = NumberSeq::newGetNum(numberSequenceReference);
this.setField(fieldnum(FMTrip, TripNumber),num.num());
}
}
```

Notes

- The **Number sequences** list page can be used to create number sequences for the references with multiple segments described in the previous scenario.
- The same approaches described in the Forms and Enterprise Portal examples in this document are used for a number sequence field exposed on a form in the client or on a page in Enterprise Portal.
- Parameters forms have not been modified to support the assignment of number sequences to references with multiple segments. The ISV or partner developer has the option of modifying the **Number sequences** page in parameters forms to provide this capability.

Change of scope for an existing reference

In this scenario, a developer is changing the scope of an existing reference. An example of this type of scenario is changing the vehicle number to be global (or shared). The existing reference, **VehicleNum**, has a scope of DataArea segment, but the developer wants to change this to a global scope.

This scenario is based on the Company scenario, but a few code changes are required to accomplish the change of scope. Note that in this scenario you do not need to create the **Number sequences** page in the Parameters form of the module because it already exists.

To implement a change of scope:

1. Create a new enum value **FM** with label **Fleet Management** in base ENUM `NumberSeqModule`.
2. This step is nearly identical to Step 2 in the Company scenario, but removes the segment of DataArea for **VehicleNum**. The sample code for creating the class is as follows.

```
Class declaration

public class NumberSeqModuleFleetManagement extends
  NumberSeqApplicationModule
{

}

protected void loadModule()
{
    NumberSeqDatatype datatype = NumberSeqDatatype::construct();

    /* Vehicle Number */

    datatype.parmDatatypeId(extendedtypenum(VehicleNum));
    datatype.parmReferenceHelp("Unique key for Fleet Management
  vehicles");
    datatype.parmWizardIsContinuous(false);
    datatype.parmWizardIsManual(NoYes::No);
    datatype.parmWizardIsChangeDownAllowed(NoYes::No);
    datatype.parmWizardIsChangeUpAllowed(NoYes::No);
    datatype.parmWizardHighest(999999);
          this.create(datatype);

    /* Trip Number */

    datatype.parmDatatypeId(extendedtypenum(TripNum));
    datatype.parmReferenceHelp("Unique key for trips");
    datatype.parmWizardIsContinuous(false);
    datatype.parmWizardIsManual(NoYes::No);
    datatype.parmWizardIsChangeDownAllowed(NoYes::No);
    datatype.parmWizardIsChangeUpAllowed(NoYes::No);
    datatype.parmWizardHighest(999999);

    datatype.addParameterType(NumberSeqParameterType::DataArea, true,
  false);
    this.create(datatype);

}

public NumberSeqModule numberSeqModule()
{
    return NumberSeqModule::FM;
```

3. Instantiate the new global scope for a vehicle number. To use that scope in the API call to the framework to generate a new vehicle number:
   - Add a method on the parameter table (**FleetManagementParameters**) for the new module.

```
Public server static NumberSequenceReference numRefVehicleNumber()
  {

NumberSeqScopeFactory::CreateGlobalScope();
 return NumberSeqReference::findReference(extendedtypenum
  (VehicleNumber));
    }
```

   - Add a table method `FMVehicle::setVehicleNumber` for creating a new vehicle number.

```
Void setVehicleNumber()

{
NumberSeq num
NumberSequenceReference numberSequenceReference;

numberSequenceReference =
  FleetManagementParameters::numRefVehicleNumber();
if(numberSequenceReference)
{
num = NumberSeq::newGetNum(numberSequenceReference);
this.setField(fieldnum(FMVehicle, VehicleNumber),num.num());
}
}
```

Note The same approaches described in the Forms and Enterprise Portal examples in this document are used for a number sequence field exposed on a form in the rich client or on a page in Enterprise Portal.

Code upgrade

Code upgrade scenarios are used when ISV developers need to upgrade the code for existing number sequences. Code upgrades are performed when there has been a change in the definition of a reference that allows for a different scope to make use of the new capabilities of the framework. Another common scenario occurs when partners and customer IT system developers need to upgrade the code for number sequences created through customizations.

The following examples describe scenarios in which code changes are required:

- The objects identified in an extensibility scenario, such as the enum `NumberSeqModule` and the class `NumberSeqModuleXXX`, must be upgraded. Because the name of the class has also changed, you need to change the name of the

subclass. For example, a NumberSeqModuleXXX subclass, such as NumberSeqModuleFleetManagement, previously took the format of a **NumberSeqReference_XXX** class, and was named NumberSeqReference_FleetManagement. In addition, in a NumberSeqModuleXXX class, the LoadModule method must be modified to include a DataArea segment, as follows:

```
datatype.addParameterType(NumberSeqParameterType::DataArea, true, false);
```

The **NumberSeq** API that uses a segment of Company (or DataArea) to generate a number sequence has not changed and there is no need to change any code that calls the API.

If your code pattern is to call the API using a number sequence code or ID, you will need to make changes to correct the table relations. The use of RecId as the primary key has caused changes to the **NumberSequenceTable**, which requires you to update the primary key-foreign key relations on any application table you might have in your customization or solution.

For example, the table relation **NumberSequenceTable** was the following on the **kanbanRule** table:

```
kanbanRule.CardsNumberSequence == NumberSequenceTable.NumberSequenceCode
kanbanRule.DataAreaId == NumberSequenceTable.DataAreaId
```

And it was changed as follows:

```
kanbanRule.CardsNumberSequence == NumberSequenceTable.RecId
```

The following table provides a comparison of the code changes between Microsoft Dynamics AX 2009 and Microsoft Dynamics AX 2012.

Code Pattern	**Object type**	Microsoft Dynamics **AX 2009**		Microsoft Dynamics **AX 2012**	
		Value or instance	Example	Value	Example
Number Sequence Reference definitions	Class	NumberSeqReference_XXX	NumberSeqReference_FleetManagement	NumberSeqModuleXXX	NumberSeqModuleFleetManagement

	Method	LoadModule	The method changed in Microsoft Dynamics AX 2012 with an additional statement to define scope of the extended data type (EDT), and statements to set the attributes of the EDT.	Loadmodule	`datatype.addParameterType(NumberSeqParameterType::DataArea, true, false);`
					
					`datatype.parmDatatypeId(extendedtypenum(VehicleNum));`

API usage based on ID or code	Relation	Relation Instance - NumberSequenceTable	`kanbanRule.CardsNumberSequence == NumberSequenceTable.NumberSequenceCode kanbanRule.DataAreaId == NumberSequenceTable.DataAreaId`	Relation instance	`KanbanRule.CardsNumberSequence == NumberSequenceTable.RecId`

Data upgrade

If there are changes to the configuration of an extended data type, or reference, you will need to modify the data upgrade scripts to reflect those changes. For example, suppose the reference for a vehicle number had been based on the DataArea segment in Microsoft Dynamics AX 2009 but has been changed to a global number sequence in Microsoft Dynamics AX 2012. This means that the reference now has no segments.

If the format for a vehicle number is V_######, the same number, V_000001, might have been generated in multiple companies when the segment was DataArea. If the reference has been changed to a global number sequence, you will need to remove the duplicates in the data before that data can be migrated to the target system during an upgrade. This is accomplished by modifying the data upgrade script to handle the duplicates.

For information about how to modify data upgrade scripts, see the white paper "How to write data upgrade scripts for Microsoft Dynamics AX" on the Microsoft Connect site.

Microsoft Dynamics is a line of integrated, adaptable business management solutions that enables you and your people to make business decisions with greater confidence. Microsoft Dynamics works like and with familiar Microsoft software, automating and streamlining financial, customer relationship and supply chain processes in a way that helps you drive business success.

U.S. and Canada Toll Free 1-888-477-7989

Worldwide +1-701-281-6500

www.microsoft.com/dynamics