QAD Enterprise Applications
Enterprise Edition

User Guide

QAD Automation Solutions: Data Collection

Overview
Setup
Create Transactions, Events, Lookups, and Message Overrides
Create the Field Device Menu

70-3339-3.1-Rev1
Version 3.1
QAD Enterprise Applications 2018
December 2018

DataCollection_UG_v031.pdf/r9m

QAD Inc.
100 Innovation Place
Santa Barbara, California 93108
Phone (805) 566-6000
https://www.qad.com

Contents

Data Collection User Guide Change Summary

The following table summarizes significant differences between this document and previous versions.

Date/Version	Description	Reference
December 2018/v3.1-Rev1	Added a link to the Data Collection - Transaction Definition Publishing video	page 74
September 2018/v3.1	Rebranded for v3.1	--
	Removed EAM technical requirements	--
	Updated Data Collection programs table to include transaction key programs	page 10
	Updated the Reports section to include the Transaction Key Report	page 11
	Updated Data Collection Control section	page 16
	Updated Define the Program section to include new fields for class files	page 22
	Updated Import/Export Programs section to include new fields for class files	page 26
	Added issues to Troubleshooting section	page 35
	Updated Create Transactions section to include new fields for class files	page 38
	Added new section: Publishing Transactions	page 74
March 2018/v3.0	Rebranded for v3.0	--
	Added section: Enabling Effective Operations Using QAD Automation Solutions	12
October 2017/v2.1.0	Rebranded for v2.1.0	--
September 2016/v2.0.1	Rebranded for v2.0.1	--
	Changed section title from Integration with Other QAD EE Modules to Integration with Other QAD EE Programs.	8
	Added troubleshooting table.	35
	Revised Maintain Field Details section by updating the screenshot and adding the Edit Entry Event and Edit Exit Event field definitions.	51
	Removed section: Launch Data Collection in QAD .NET UI	--
March 2016/v 2.0	Initial release	--

Chapter 1

Overview

This chapter discusses the following topics:

Introduction

QAD Data Collection lets system administrators and implementers build transactions that users of radio frequency (RF) devices or other mobile field devices can process. The personalized transactions align with the user's manufacturing and materials logistics needs.

Development tools are provided with Data Collection, so system administrators can customize the RF transactions to provide the transaction flow, UI layout and fields, and data validations that the business requires. Data Collection includes a transaction library that you can use or customize to meet your business needs.

The transactions—used in receiving, production, shipping, inventory management, and more—are processed in connection with a QAD server that provides Data Collection, data validation, and error handling during transaction processing and line-by-line or unit posting of the RF transactions in real time.

The Data Collection features and functions are fully incorporated within QAD EE, removing the need for middleware interaction or processing and eliminating the possibility of transactions not being posted. Users can run Data Collection transactions on a hand-held device, like the RF device or iPhone, or in an RF simulator in .NET UI.

Data collection sources are typically the barcodes that RF devices scan. RF users can transfer, package, and ship product, and adjust QAD EE inventories using item, serialization, or pallet IDs, or other identification tags—eliminating paperwork and manual data entry. The data is delivered in real time to QAD EE and validated from the QAD database. QAD EE is updated with each RF or mobile field device transaction as it happens, so error-handling and error displays occur during the processing. There is no need for separate databases and manual data entry. This also means that accurate and real-time data is available *immediately* for reporting. This lets schedulers, planners, material expediters, sales personnel, and others have timely data regarding production processes.

The system maintains Data Collection transaction history by serial number and other identifying information such as date, employee, site, and device.

Features

Using transactions that implementers build in Data Collection for hand-held or mobile field devices lets device users:

- Reduce time required to input production data.
- Reduce data errors.
- Improve data accuracy and production efficiency.
- Deliver real-time shop floor feedback.
- Route labels to a printer.
- Maintain various QAD EA versions.
- Make data available for real-time queries, reporting, and other analysis, which improves decision making.

Specific Features

- **Record locking.** Data Collection features record-locking functions that provide visibility to the user ID and terminal with the locked record. The RF device displays record-locking and record-locking timeout messages. Also, the number of record-locking times has been minimized due to the Data Collection design, so that record-locking limits are no longer an issue.
- **Visible processed transactions.** The system informs you when records have been successfully committed and transactions were successfully or unsuccessfully processed. Data Collection bypasses QXtend and calls APIs directly, allowing for direct and immediate validation and rapid display of transaction results.
- **Importing/Exporting Transactions and Transaction Data.** You can use several Data Collection programs to import or export dataset definitions, program definitions, transaction definitions, events, and lookups, to and from any QAD EE version. The import/export programs let you specify the operation and use filter masks or select all data.
- **Reporting**. You can use programs in the Data Collection Report Menu (6.20) to report on a range of:
  - Transactions by name; version; and, optionally, details
  - Programs by name, including the function/procedure, keyword, and application version
  - Application message numbers and use message numbers, optionally sorting by either number type

Data Collection Components

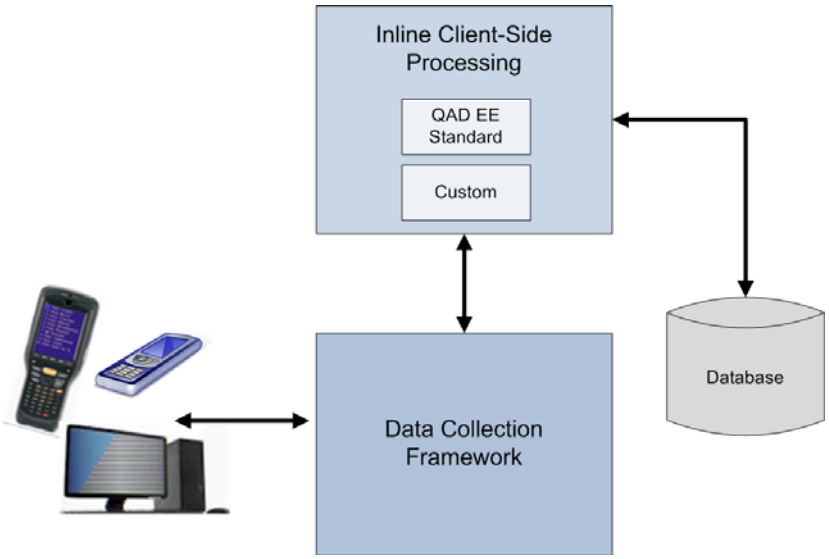
Data Collection consists of two major components:

- **RF Framework.** This is the Data Collection architecture. The architecture handles record locking, sends data sets to APIs, and manages connections to servers.

  The architecture includes the Data Collection engine. These pieces comprise the infrastructure and configuration files to execute the RF transaction.
- **Transaction Development Toolset.** This includes functions and maintenance programs that let you develop transactions for your business. Developers can import existing programs and functions, import datasets, then modify the imported data to create customized transactions for the mobile field devices. They can import from and export to any QAD EE environment, letting them customize transactions for a particular version. Finally, they can test the Data Collection directly from within the Data Collection module, shortening the test time for modified or new transactions.

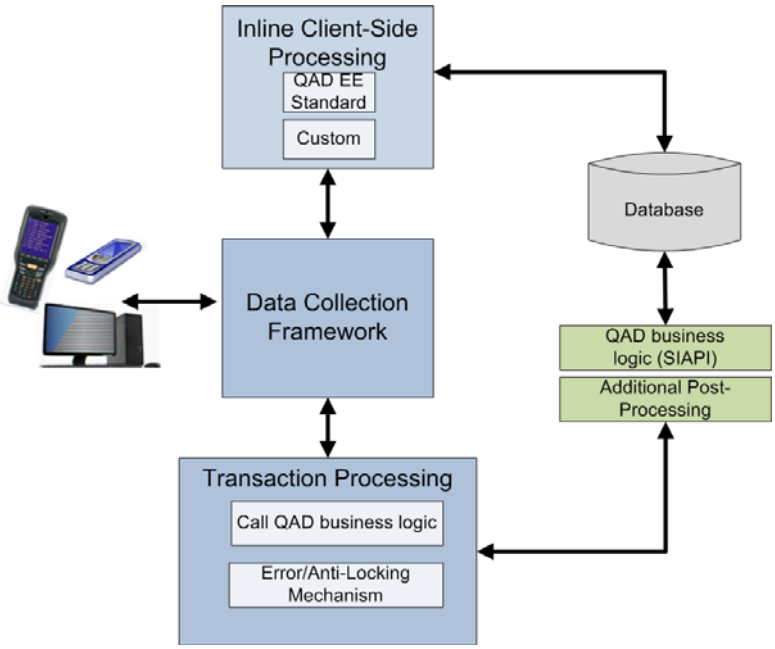
The following graphic depicts the RF framework with a front end that includes input from various devices, such as a PC running an RF emulator, an RF device, or smart phone. On the client side, components provide validations and functions by calling QAD standard programs and procedures directly from the QAD database or can plug in custom validations, such as for additional restrictions on the shop floor and procedures.

Fig. 1.1
Architecture Overview



To send the information, the architecture can call QAD business logic with additional record-locking functions and call the SI APIs. The framework can also call additional processing such as requests to send different labels.

Fig. 1.2
Architecture Overview, Expanding



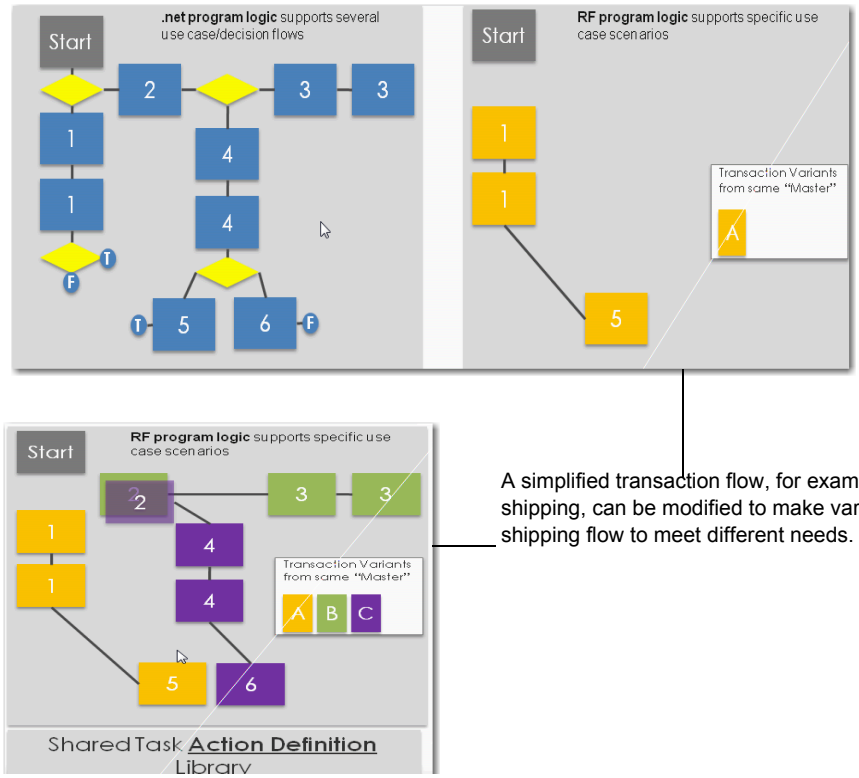
Transaction Development

The Data Collection module contains several maintenance programs that you use to set up or maintain transactions that run on a mobile field device. The ability to configure and customize the transactions that run on the devices is at the forefront of the Data Collection capabilities. This means that you can customize the way fields display and the sequence in which they display, customize fields, define field defaults, and provide drop-down lookups for fields for RF users.

You determine what business logic and events you want to call during the transaction. You can call the QAD EE standard programs and code or customized programs and code directly. This means that you do not use middleware that duplicates QAD code, which can sometimes cause issues when you upgrade, or when your users have different UIs.

Note that Data Collection does not duplicate the standard .NET UI transaction flow; instead, the transaction flow is simplified, supports specific use case scenarios, and is customizable. Because of the simplified transaction flow, RF users are not required to make decisions. They simply invoke transactions, enter data when prompted, confirm transaction completion—and, hence, transaction committal—and print labels when required.

Fig. 1.3
Simplified Transaction Flow



A simplified transaction flow, for example, for shipping, can be modified to make variations of the shipping flow to meet different needs.

When you customize the transactions, you are creating a workflow for your business so that specific transactions begin when a certain transaction ends; for example, you may want your users to build a serialized pack, and when finished, immediately transfer the pack to a specific holding area. You can carry this further, and have users prompted to print serialized labels for the pack once they complete the pack build.

For more information, refer to Chapter 2, “Setup,” on page 13.

Build Transactions

You can use Transaction Definition Maintenance (6. 16) to fully build and customize the transaction. The program lets you specify transaction details and datasets, then drill down into defining and customizing other aspects such as the pre- and post-processing events, buffers, parameters, fields, UI style, and more. You can also specify application connection data through the program. See “Create Transactions, Events, Lookups, and Message Overrides” on page 37.

The program follows a similar flow of setting up the sequence for each portion of the transaction you are defining, then drilling down to set up details of each portion. This lets you customize the way that transactions present screens to RF users, when they enter data, when processing occurs, when committal occurs, and so on.

Clone Transactions

You can clone Data Collection transactions for rapid development, personalization, and duplication.

Cloning a transaction, such as an inventory transfer transaction, then creating a variant of the transaction is an easy method to customize a transaction. The variants are based on the source version, but remain unique with the customizations you make.

Event Creation

You can control the events and the timing of events; for example, you can determine whether an in-line validation occurs, such as verifying whether an item exists before committing the transaction or having custom default locations display while processing. For more information, see “Create Events” on page 56.

You can control when you want pre-processing or post-processing events to occur. You also have full control over the behavior of the event. For example, you can change the flow for your processes and determine when users post, when screens and processes loop, and so on; for instance, you can determine whether transactions are committed to the database as a unit or on a line-by-line basis or based on an event, such as committing after every quantity decrease. Event types include the following:

- In-line custom
- In-line standard duplication
- In-line standard direct calls
- QAD API processing

Create Templates

When you create a transaction, you can save time by using templates for events or lookups.

During the creation of the transaction when you create an event or a lookup, you can optionally save your event and lookup creations to templates, then reuse the templates later in other transactions that you build.

You can also use Event Template Maintenance (6.22.3) to set up a template for programs, functions, and so on that the system calls as an event process in other transactions that you build. You can do the same with lookups; that is, you can use Lookup Template Maintenance (6.22.4) to build lookup templates.

Export/Import Transaction and Transaction Data

You can use several Data Collection programs to import or export the following to and from any QAD EE version:

- Dataset definitions
- Program definitions
- Transaction definitions
- Events
- Lookups

The import/export programs let you specify the operation, use filter masks or select all data, or specify to overwrite files. For more information, see “Import/Export Programs” on page 26.

Getting Data Collection to Work

Data Collection is a toolset designed for application developers and application implementers. It lets you develop transactions, based on QAD EE data and programs, then run the transactions—typically with an easier user interface, fewer keystrokes, and other new abilities like record locking—on mobile field devices, like the RF device.

You can use the dataset definition, program definition, and program version of your choice. Your objective is to define the dataset to use, define the programs that use the data, then build transactions based on the data and programs so that mobile field devices can run the transactions based on the environment you selected.

To that end, you can use a dataset definition that exists anywhere, so you must import the dataset into the environment you want to use to build the transactions. Likewise, you import the program definitions. You can edit both the dataset and programs once you import them using maintenance programs within the Data Collection module. Also, you can either import transactions, then edit them using the Transaction Definition Maintenance, or create new ones in the maintenance program.

The transactions can have events associated with them. You can use an event template, or create the events when you create the transactions then save them to a template for later use. You can associate the events with datasets, buffers (tables), fields, or lookups, and you can set them up for pre-processing, post-processing, or commit processing. You can define the event method, then set up parameter details for the event. You can also build lookups. You can import/export lookups, and save them as templates for use in other transactions.

Once you build the transactions, events, and lookups, you set up the Mobile Device menu elements using Data Collection programs, then test the menu and test the Data Collection from the transactions that you built by running the actual mobile device menu directly from the Data Collection module. See “Create the Field Device Menu” on page 89.

Integration with Other QAD EE Programs

Data Collection interfaces with the following QAD programs:

- Serialization
- Label Printing Services (LPS)
- EAM

Serialization

QAD's Serialization module provides a framework for tracking and tracing both lot numbers and serial IDs independently, but linking them together where both are used. In addition to the item serialization capabilities, the serialization solution provides for license plate inventory management capabilities, letting you uniquely identify packaging units.

Serialization users can define any type of packaging unit—boxes, cases, or pallets, for example—and any number of packaging levels. They can define packaging structures when they receive product per item, origin, or destination; assign label formats by packaging type, origin, or destination to use when printing labels, and attach label IDs to packs to identify product. In addition to defining serial number format IDs and managing packaged inventory by identity, serialization users can communicate serialized data externally through EDI with customers or trading partners.

So, when RF users run the Data Collection-provided or customized transaction, they can scan a single serial number so that the item, quantity, lot/serial, weight, and other attributes are automatically associated with that transaction without additional scans. Data Collection validates all information, including serialized data, for accuracy, and the system updates each transaction so that real-time updates exist in the QAD EE database. The system updates transactions using your configured rules. Using Data Collection features and functions, RF users can simply log in, select a category and a category process, then select the transaction to run, such as a transfer or a cycle count, and begin processing without repeatedly entering the serialized numbers.

Label Printing Services

You can also print serialized and non-serialized barcode labels for items, containers, and mixed pallets anywhere in the inventory tracking process, using QAD EE Label Printing Services functions from the RF device for the RF transactions. The Data Collection package includes predefined bar code label formats that are integrated with each transaction.

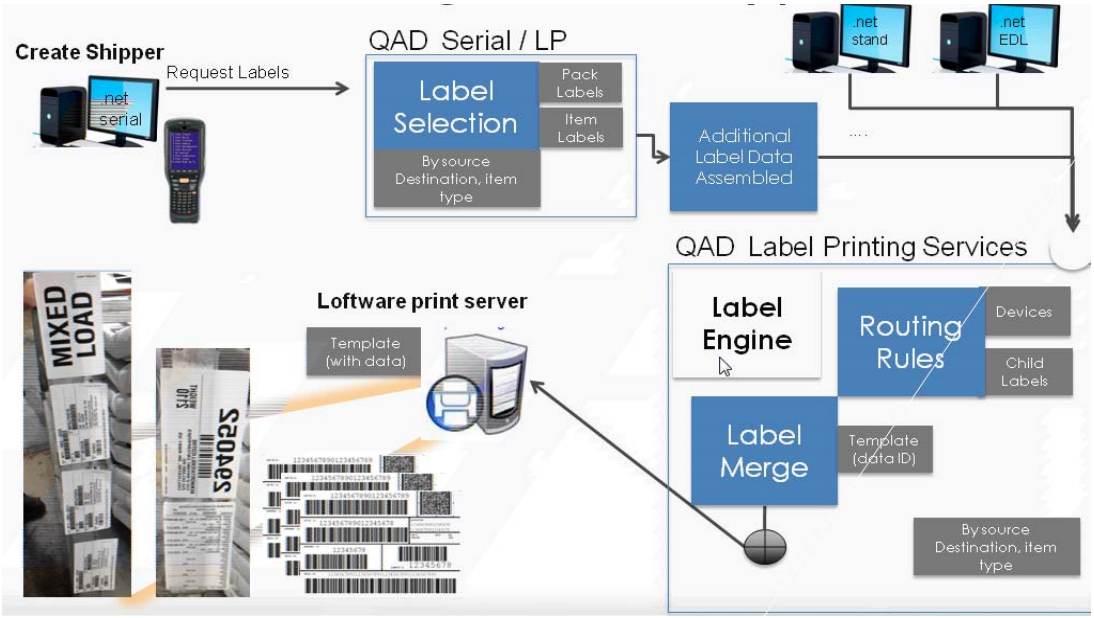
Data Collection uses QAD EE Label Printing Services data, which eliminates the need for RF users to manually enter data or use outside label databases. You can customize label formats or produce your own labels to meet specific label requirements.

Using QAD Label Printing Services functions, you can direct where the label prints based on user ID, label application, device, or content. QAD Label Printing Services considers:

- The packaging required—the pack or item—for the labels
- Source, destination, and item type
- Additional label requirements such as the shipping address or EDI requirements because RF users are not required to enter this data when printing labels

QAD EE Label Printing Services determines the printer, when to merge labels, and more. The QAD Label Printing Services functions are called directly by the hand-held device UI or emulator on the PC from the framework, or if you use Serialization functions, they call Label Printing Services functions directly; see Figure 1.4. Label Printing Services sends the request to third-party print server (Loftware). For more information, refer to *QAD Label Printing Services User Guide*.

Fig. 1.4
Label Flow



EAM

QAD Enterprise Asset Management (EAM) allows you to effectively manage your organization's physical assets. The EAM suite of applications maximizes manufacturing equipment use and minimizes repair costs. Lifecycle management includes design, construction, commissioning, operation, maintenance, and replacement of plant, equipment, and facilities. EAM provides supply chain management solutions for project accounting, plant maintenance, and maintenance, repair, and operation (MRO) including inventory and purchasing. EAM is fully integrated with the Manufacturing and Financial modules in QAD EE.

The following transactions have been added to Data Collection to allow you to perform essential EAM functions:

EAM Module	Transaction Type
Inventory	Issue to WO
	Issue to Equipment
	Issue to Project/Job
	Issue to CC/Acct
	Return
	Relocate
	Adjust
	Part Look-up
Physical Inventory	Count Physical Inventory

EAM Module	Transaction Type
Purchasing	PO Receipt
Stores Requisition	Issue to Stores Requisition
	Close Stores Requisition
	Part Look-up
Work Orders	Post Labor to WO
	Post Downtime to WO

In addition to these transactions, EAM part and part location labels are provided in Data Collection.

Data Collection Programs

The following table lists the programs that comprise the Data Collection functionality:

Table 1.1
Data Collection Menu Programs

Menu	Menu Label	Program Name
6.	**Data Collection Menu...**	
6.1	Gather Data	dclaunch.p
6.12	Process Category Maintenance	dccatmt.p
6.13	Task Type Maintenance	ctktpmt.p
6.14	Dataset Definition Maintenance	dcdsdfmt.p
6.15	Program Definition Maintenance	dcpgdfmt.p
6.16	Transaction Definition Maintenance	dctranmt.p
6.17	Process/Task/Definition Xref	dctkacmt.p
6.20	**Reports Menu....**	
6.20.1	Transaction Definition Report	dctranrp.p
6.20.3	Program Definition Report	dcpgdfrp.p
6.20.5	Message Override Report	dcmsgrp.p
6.20.13	Transaction Key Report	NA
6.22	**Configuration Menu....**	
6.22.1	Service Connection Maintenance	dcconmt.p
6.22.2	Application Version Maintenance	dcvermt.p
6.22.3	Event Template Maintenance	dcevtpmt.p
6.22.4	Lookup Table Maintenance	dclktpmt.p
6.22.13	Message Override Maintenance	dcmsgmt.p
6.22.14	Device Maintenance	dcdevmt.p
6.22.17	Transaction Key Maintenance	dclicensemt.p
6.22.24	Data Collection Control	dcdcuipm.p
6.23	**Data Collection Utilities Menu.....**	
6.23.1	Process Category Import/Export	dccatgie.p
6.23.2	Program Definition Import/Export	dcprgmie.p
6.23.3	Dataset Definition Import/Export	dcdsdfie.p

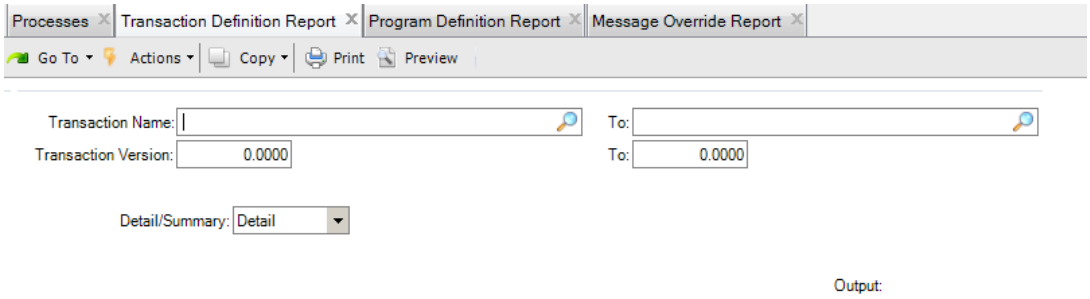
Menu	Menu Label	Program Name
6.23.4	Transaction Definition Import/Export	dctranie.p
6.23.5	**Event Template** Import/Export	**dcevtpie.p**
6.23.6	Lookup Template Import/Export	dclktpie.p
6.23.7	Message Override Import/Export	dcmsgie.p
6.23.23	Transaction Key Import/Export	dclicenseie.p

Reports

You can use any of the following reports in the Data Collection Reports menu (6.20):

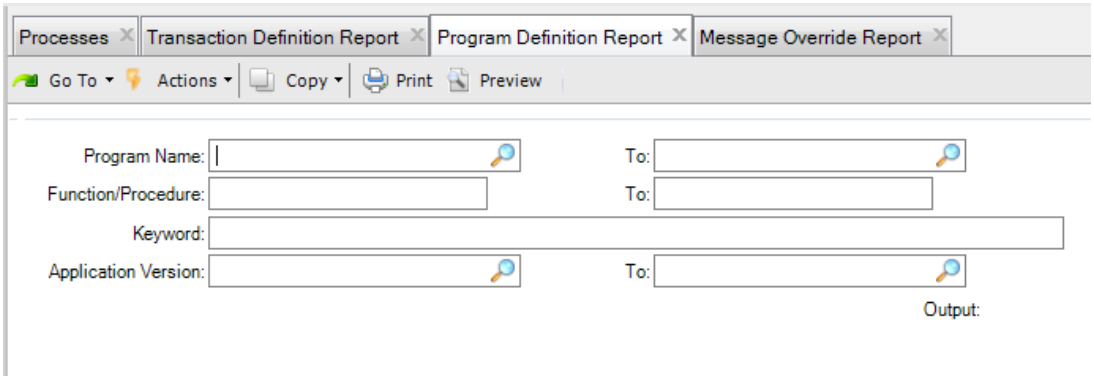
- **Transaction Definition Report (6.20.1).** Enter a range of transaction numbers or version numbers, then specify whether the system displays details.

Fig. 1.5
Transaction Definition Report



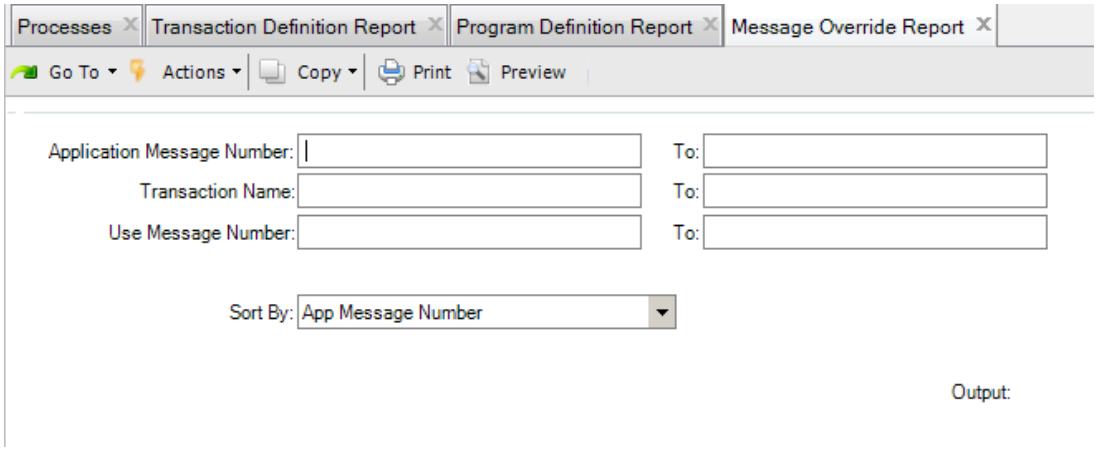
- **Program Definition Report (6.20.3).** Enter a range of program names, function/procedures, or application versions or specify keywords, separated by commas.

Fig. 1.6
Program Definition Report



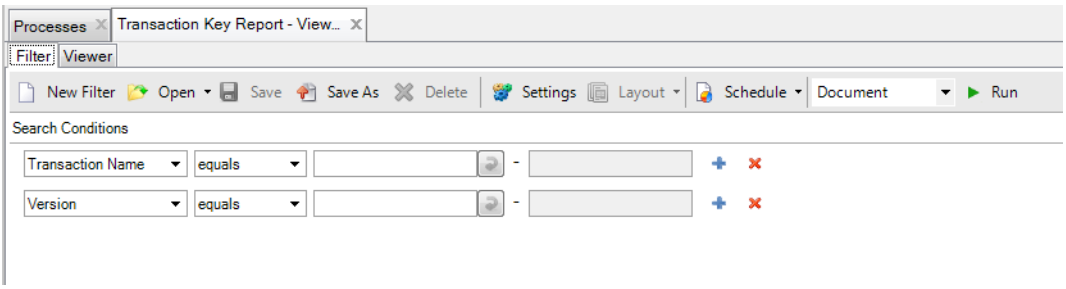
- **Message Override Report (6.20.5).** Specify a range of application message numbers, transaction names, or use message numbers; then, sort by application message number or use message number.

Fig. 1.7
Message Override Report



- **Transaction Key Report (6.20.13).** Shows the transaction keys for each transaction. This report also shows every transaction name and version that has been in the system. For more information see, “Transaction Key Report” on page 79.

Fig. 1.8
Transaction Key Report



Enabling Effective Operations Using QAD Automation Solutions

Watch the following videos for an in-depth overview about how QAD Automation Solutions can enable you to run effective operations:

- Bending ERP
  https://media.qad.com/mediasite/Play/818a9627fa3545a7863b55cdeba09ac41d
- Inbound Materials
  https://media.qad.com/mediasite/Play/6dbcc53a54724f9caea5e3d3620893891d
- Production
  https://media.qad.com/mediasite/Play/9e9a2fa0219f4737bfc810d8b98cc7221d
- Outbound Materials
  https://media.qad.com/mediasite/Play/f288dca6ed7946728a34e445f21d7d2b1d

Chapter 2

Setup

This chapter discusses the following topics:

Setup Overview

The Data Collection module contains several maintenance programs that you use to:

- Set controls, versions, and connections.
- Create/modify dataset and program definitions.
- Create transactions and events.
- Import/export definitions, programs, and more to and from other environments.
- Create the mobile device menu elements.
- Test the mobile device menu and collect data from the menu transactions.

What You Need to Know

Data Collection is a development tool. Implementers who use the tool need to know:

- What the APIs do and the data the APIs expect
- The programs to use
- The dataset structure, so that you can gather all information to build the transactions

Basic Flow to Set Up and Use Data Collection

Data Collection lets you use the dataset definition, program definition, and program version of your choice. Your objective is to define the dataset to use, define the programs that use the data, then build transactions based on the dataset and program definitions for the environment of your choice so that mobile field devices can run the transactions that you build.

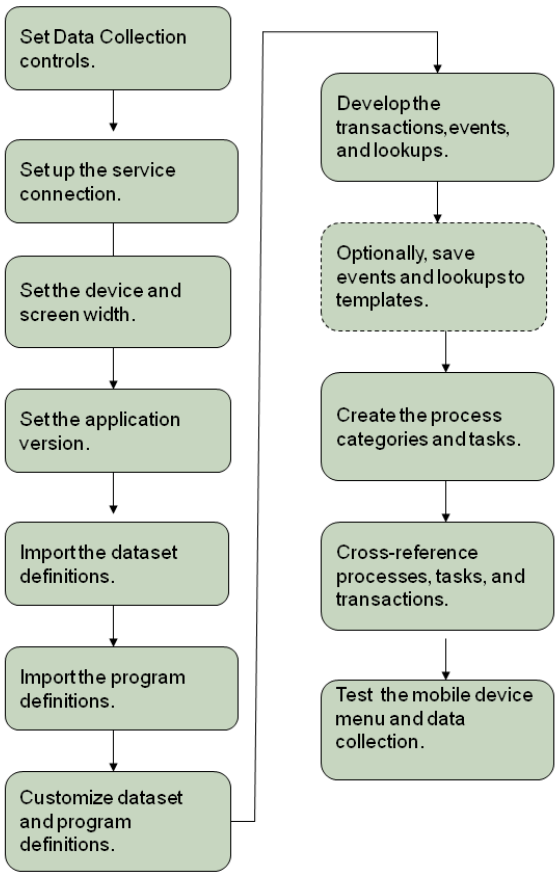
To that end, you can use a dataset definition that exists anywhere, so you must import the dataset into the environment you want to use to build the transactions. Likewise, you import the program definitions. You can edit the dataset and program definitions using the Data Collection maintenance programs.

You can either import transactions, then edit them using a Data Collection maintenance program, or create new ones in the maintenance program. The transactions can have events associated with them. You can associate the events with the dataset, buffers (tables), fields, and lookups. You can define the event method, then set up parameter details for the events.

Once you build the transactions, events, and lookups, you set up the Mobile Device menu elements using Data Collection programs, then test the menu and collect data from the transactions that run in the menu directly from the Data Collection menu.

Use the following graphic as a guideline to the basic setup flow for Data Collection.

Fig. 2.1
Setup Flow



Use the information in Table 2.1 to help you find the programs to use for each step in the flow:

Table 2.1
Step/Program Cross-Reference

Step in Flow	Link to Chapter /Heading	Data Collection Program to Use
Set up Data Collection controls.	“Data Collection Control” on page 16	Data Collection Control (6.22.24)
Set up the service connection.	“Service Connection Maintenance” on page 18	Service Connection Maintenance (6.22.1)
Specify the device and device screen width.	“Device Maintenance” on page 20	Device Maintenance (6.22.14)
Set the application version.	“Application Version Maintenance” on page 21	Application Version Maintenance (6.22.2)
Import the dataset definition.	.“Define the Dataset” on page 21	Dataset Definition Import/Export (6.23.4)
Import the program definition.	“Define the Program” on page 22	Program Definition Import/Export (6.23.2)
Customize the dataset definitions and program definitions	“Define the Dataset” on page 21 “Define the Program” on page 22	Dataset Definition Maintenance (6.14) Program Definition Maintenance (6.15)
Develop the transactions, events, and lookups.	“Create Transactions” on page 38	Transaction Definition Maintenance (6.16)

Step in Flow	Link to Chapter /Heading	Data Collection Program to Use
Optionally save events and lookups to templates.	“Create Event Templates” on page 66	Event Template Maintenance (6.23.3)
	“Create Lookup Templates” on page 73	Lookup Template Maintenance (6.23.4)
Create the process categories and tasks.	“Create Process Categories” on page 90	Process Category Maintenance (6.12)
Cross-reference processes, tasks, and transactions.	“Create a Task and Category/Process Cross-Reference” on page 92.	Process/Task/Definition Xref (6.17)
Test/view the mobile device menu and Data Collection.	“Gather Data” on page 94	Gather Data(6.1)

Basic Setup

You use the programs in the Data Collection Configuration Menu (6.22) to set up Data Collection:

- Data Collection Control
- Service Connection Maintenance
- Device Maintenance

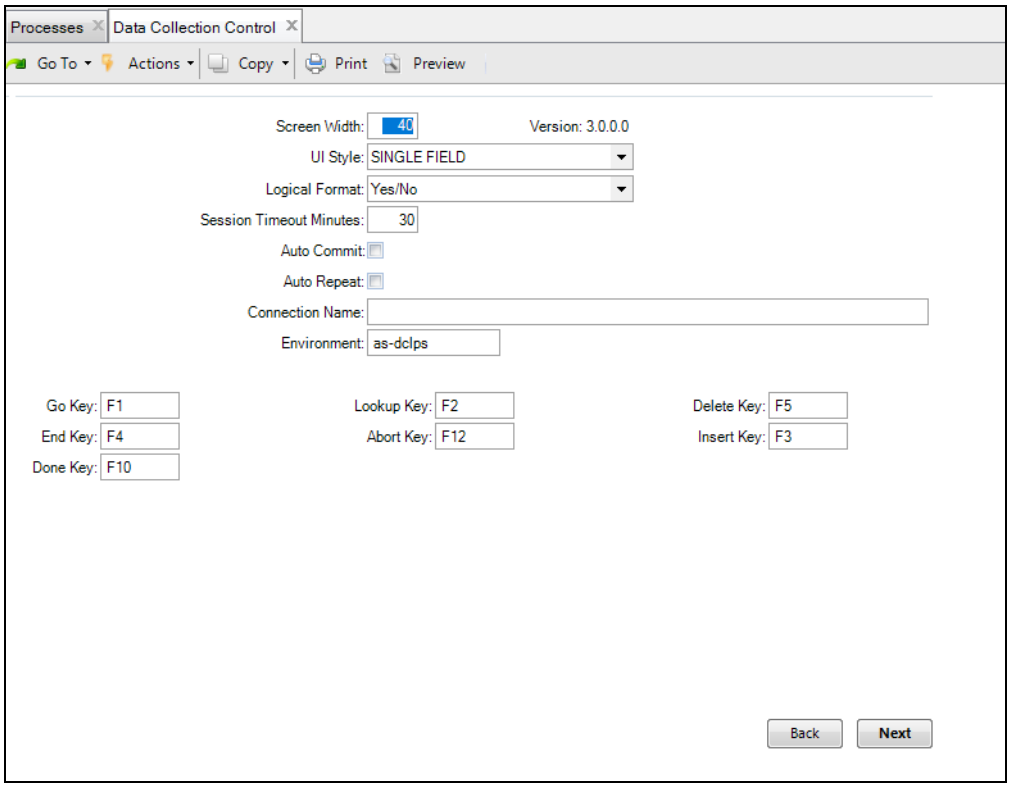
Note Other programs exist within the Configuration menu; however, you use these programs to create templates or override messages; see “Create Transactions, Events, Lookups, and Message Overrides” on page 37.

Data Collection Control

Use Data Collection Control (6.22.24) to set controls for using functions and features in the Data Collection module.

When you enter values in the control file field, the system populates the same-named field in other Data Collection programs.

Fig. 2.2
Data Collection Control



Screen Width. Specify the width of the screen on which the data is being displayed.

UI Style. Select the style for the UI of the mobile device that collects data:

- **Multiple Fields.** Multiple fields display on the UI.
- **Single Fields.** Only a single-field update rows display on the UI. This is useful when the UI is smaller.

Logical Format. Specify the logical format in which RF users respond to logical prompts for Yes/No. Choose from:

- 1/0
- T/F
- True/False
- Y/N
- Yes/No

Session Timeout Minutes. Enter the time in minutes before the system automatically logs off the user for inactivity. The auto log-off functionality is only applied when the user is at the menu selection. If the user is working with a transaction, the auto log-off functionality is not activated.

Auto Commit. Specify Yes to automatically commit the transaction without prompting the mobile device user for confirmation. The default is No.

Auto Repeat. Specify Yes to automatically repeat on repeating buffers without prompting the mobile device user for confirmation. For example, during a PO receipt transaction, the mobile device user can enter several PO numbers without the system prompting the user to enter a PO number.

Note When you set this field to Yes and Auto Commit to Yes, the mobile device user can enter many POs without additional prompting, then commit the receipt transaction once.

Connection Name. Specify the connection name that is defaulted to the Connection Name Field of Event frame when creating new events.

Environment. Specify the environment.

Go Key. Enter a one-key function key on the mobile field device for the Go function. For example, enter `F1` as the function key the user presses on an RF device as the Go key. You cannot enter key combinations, such as CTRL + 3, or use invalid function key names. When you use letters, numbers, or other keys for this function, each time the mobile device user enters that key when entering data, the user invokes the function rather than the key value.

End Key. Enter a one-key function key on the mobile field device for the end function.

Abort Key. Enter a one-key function key on the mobile field device for the abort function.

Lookup Key. Enter a one-key function key on the mobile field device for the lookup function.

Done Key. Enter a one-key function key on the mobile field device for the done function.

Delete Key. Enter a one-key function key on the mobile field device for the delete function.

Insert Key. Enter a one-key function key on the mobile field device for the insert function.

Service Connection Maintenance

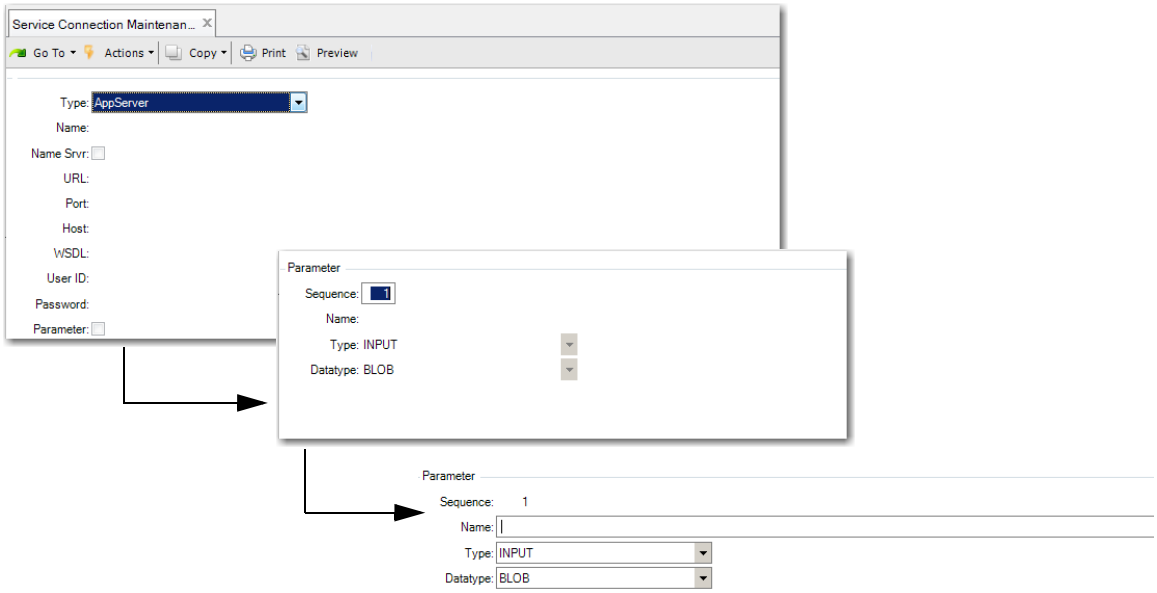
Use Service Connection Maintenance (6.22.1) to specify the connection to the Appserver. Generally, when mobile device users commit a transaction, they make a call to the Appserver. You enter the connection type, followed by additional data about the connection; for example, if you enter an HTTP connection type, you must also enter the URL.

Once you specify the connection information, enter the user ID and password for the user of the connection.

When you set Parameter to Yes, the system displays additional frames to edit the sequence for the parameters, then parameter details.

When deleting a service connection, the deletion should be active when the focus is on the URL, Host, WSDL, User, or Password fields.

Fig. 2.3
Service Connection Maintenance



Type. Enter the type of connection to the QAD server. Choose from:

- **AppServer.** Application server connection.
- **COM.** Communication interface, such as RS232.
- **Client/Local.** The client, or local machine, when connecting to a host. This requires no setup.
- **FTP.** File Transfer Protocol address to transfer files.
- **HTTP.** Hypertext Transfer Protocol for the Web.
- **SCRIPT.** The name of a script used to connect; for example, an AcquireConnection that casts the connection.
- **TCP.** Transmission Control Protocol, a protocol within the Internet protocol suite.
- **WebService.** A web hosting service, and Internet hosting service that makes a website accessible via the World Wide Web.

Name. Enter the name of a valid AppServer service that exists in AppServer Service Maintenance (36.19.1).

Name Srvr. Specify whether the connection type in use is the NameServer. The default is No.

URL . Enter the URL, if applicable. For example, if you select HTTP for Web connection, enter the URL.

Port. Enter the computer port for connection as a connection parameter.

Host. The name of the host (server) to which you are connecting.

WSDL. Enter the name of the Web Service Definition Language for XML-formatted network service endpoints.

User ID. Enter a valid user ID to use when connecting to this connection service.

Password. Enter the password for the connection service.

Parameter. Specify Yes when parameters are available for this connection type.

Sequence. Enter the sequence number in which the system passes parameters to the program, function, or procedure.

Name. Enter the name of parameter that the system passes for the connection record.

Type. Enter the parameter type. Choose from INPUT, INPUT-OUTPUT, or OUTPUT buffer.

Datatype. Enter the dataset name. Choose from:

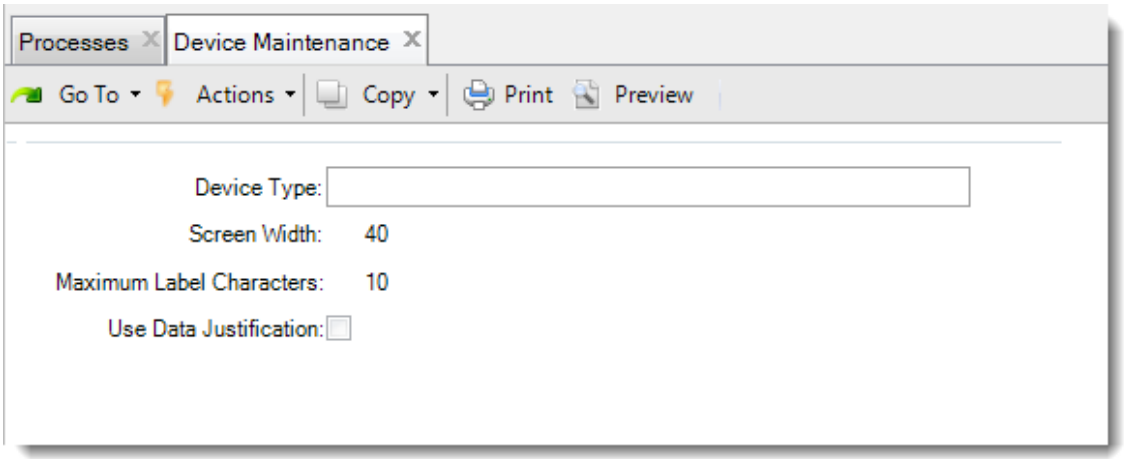
- BLOB
- CHARACTER
- CLOB
- DATASET
- DATASET-HANDLE
- DATE
- DATETIME
- DATETIME-TZ
- DECIMAL
- HANDLE
- INT64
- INTEGER
- LOGICAL
- LONGCHAR
- MEMPTR
- N/A
- RAW
- RECID
- ROWID
- TABLE
- TABLE_HANDLE
- WIDGET-HANDLE

Device Maintenance

Use Device Maintenance (6.22.14) to specify the device type and screen width of the device that runs the transactions.

During login, when the device ID is not available from the environment variable during login, the system now sets the device type as the device the user specifies in Device Maintenance.

Fig. 2.4
Device Maintenance (6.22.14)



Device Type. Enter the type of device.

Screen Width. Enter the screen width in inches.

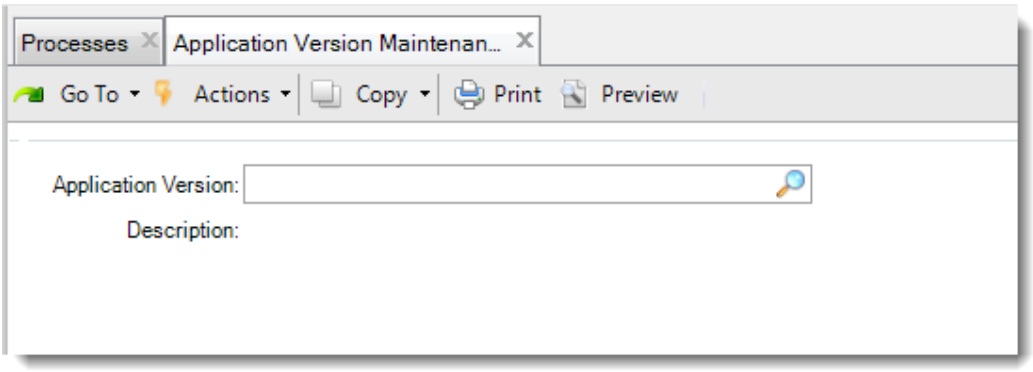
Maximum Label Characters. Specify the maximum number of characters to display for a short label. The default value is 10. This value cannot exceed: (Screen Width value - 4).

Use Data Justification. Select this check box to justify numeric data to the right.

Application Version Maintenance

Use Application Version Maintenance (6.22.2) to define the application version that the Data Collection module uses. You can update the version and have the system store version data in Data Collection version records.

Fig. 2.5
Application Version Maintenance



Version. Enter the version identifier; for example, enter QAD 2012.1EE or QAD EAM 12.5.2. This field cannot be blank.

Description. Optionally, enter a description of the application.

Define the Dataset

Use Dataset Definition Maintenance (6.14) to define a dataset definition for a particular environment. For example, you can use a dataset for a PO receipt for the 2012.1 EE environment.

Important You import the dataset to the environment first using “Import/Export Programs” on page 26.

Navigation

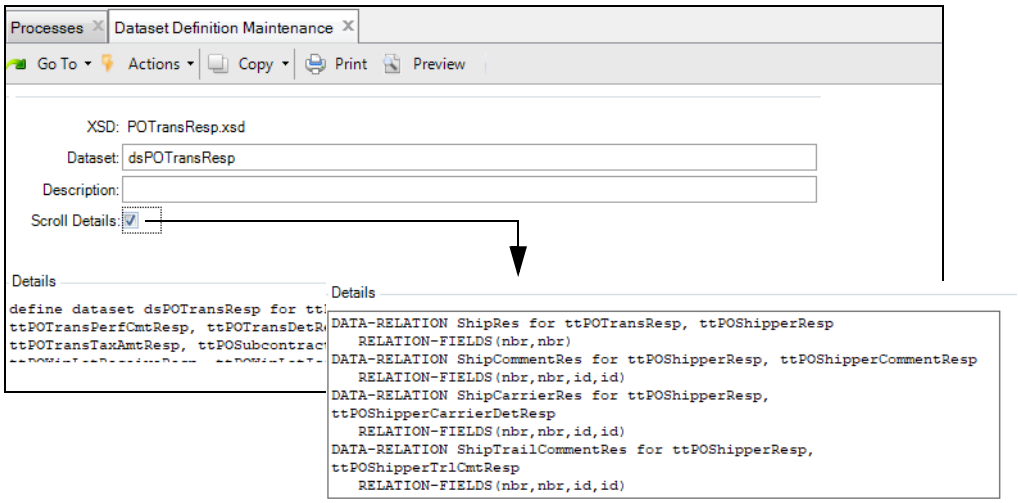
You enter the XSD name and application version. You can select an existing XSD and application version by using the drop-down browses on the fields. The system displays an error if the XSD or version is invalid or does not exist.

Optionally enter a description, then select Yes in Scroll Details to view the entire XSD. The system displays the data when you press Go. You can move the cursor up and down to view data.

Note You can enter data within the Details frame; however, the system does not save the data.

You can delete the dataset definition by putting the cursor on the Dataset field and pressing the Delete key. When multiple records exist for the Data Collection version link for an XSD file with different versions, the system deletes the record of the Data Collection version link for the XSD file and version you enter in this program.

Fig. 2.6
Dataset Definition Maintenance



XSD. Enter the name of the XSD file. You cannot leave this field blank.

Dataset. Enter the name of the dataset for the XSD file. The default is the name you enter for the API XSD file.

Description. Optionally, enter a short description of the dataset.

Scroll Data. Select Yes to view details of the dataset. The system does not keep any changes that you make to the dataset.

Define the Program

Use Program Definition Maintenance (6.15) to define a new program or modify an existing program definition that you imported into the version (environment) of your choice. The system uses the data you enter to validate and execute predefined logic for transactions you develop to run on a mobile field device or to call an API for performing transactions using QAD's Data Collection module.

You define each of the procedures and functions that are available for the program, along with the required parameters that are passed. You can define a program's specific qualities, attributes, and parameters as well as specify when the program runs persistently. You can also specify whether the program runs over an AppServer, or whether the program's procedures run asynchronously or synchronously.

You can delete the program definition by putting the cursor on the Persistent, Keyword, Description, Edit Parameter, or Notes fields and pressing the Delete key.

Navigation

1. Specify the program name in the Program field, using a program with a `.p` extension, such as `dcengxr.p`.
2. Optionally, specify the directory structure of the program. This field can be left blank.

3 Specify the QAD EE version in the Version field; then press Enter.

The system prompts to add a version link for all existing details and parameters. When you specify Yes, the system creates version link records for program details and parameters.

Note You can verify by scrolling through the program detail and program parameter records.

4 Press Enter to accept the program name and enable fields in the frame.

5 Specify the Type; Keyword; Return Data; and Edit Parameter; and optionally, a description/notes.

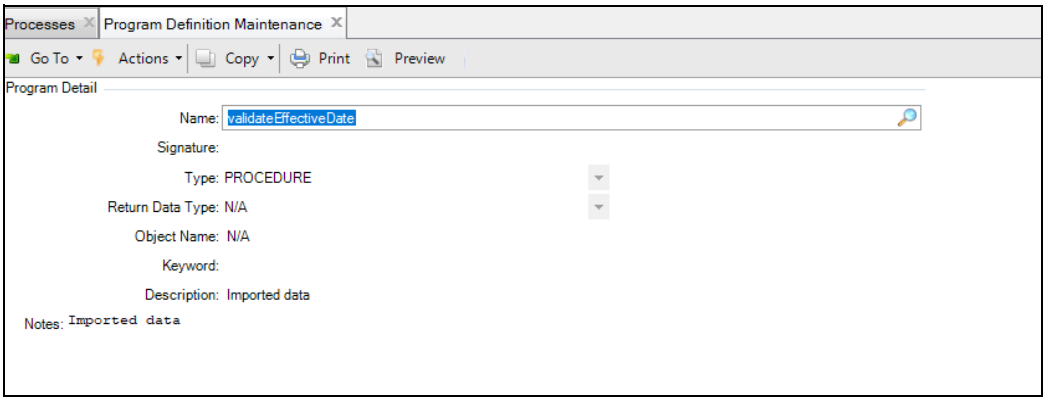
Note When Edit Parameter is Yes, the system displays the Parameter Detail Frame including the sequence, name, type, datatype, and other data. For fields in this frame, see "Parameter Detail Frame" on page 25.

The system displays the Program Detail Frame with procedures and other program functions specified by name and type.

6 Press Enter to specify parameter details, including the sequence, name, type, datatype, and other data for the Program Detail record.

For fields in this frame, see "Program Detail Frame" on page 24.

Fig. 2.7
Program Definition Maintenance



Program Name. Enter the name of the application program/source code/class file for use in the transaction. You cannot leave this field blank.

Directory Structure. Enter the path to the file on the disk that you want to run. When this field is blank, it is assumed that the directory structure is `us/<xx>`, where <*xx*> represents the first two characters of the file. For example, the directory structure for `sosomt.p` is `us/so`. In this example, this field would be left blank.

An example where this field would be populated is for `APIDispatcher.p` because the directory structure for this program is `com/qad/erp/api.` In this example, you would enter `com/qad/erp/api` in this field.

Note When a class file is selected, this field is populated with the directory structure that was entered in Program Definition Import/Export (6.23.2).

Class File. Indicates if the file selected in the Program Name field is a class file. This field is display only. For more information about class files, please see the documentation provided by Progress.

Persistent. Enter Yes when the program you defined in the Program Name field runs persistently. For new records, the default is No.

Note When the file is a class file, this field is not editable.

Keyword. Enter key words to describe this program so that you can easily find the data associated with the keyword—such as an item number—when you run reports.

Description. Enter a short description to display on reports.

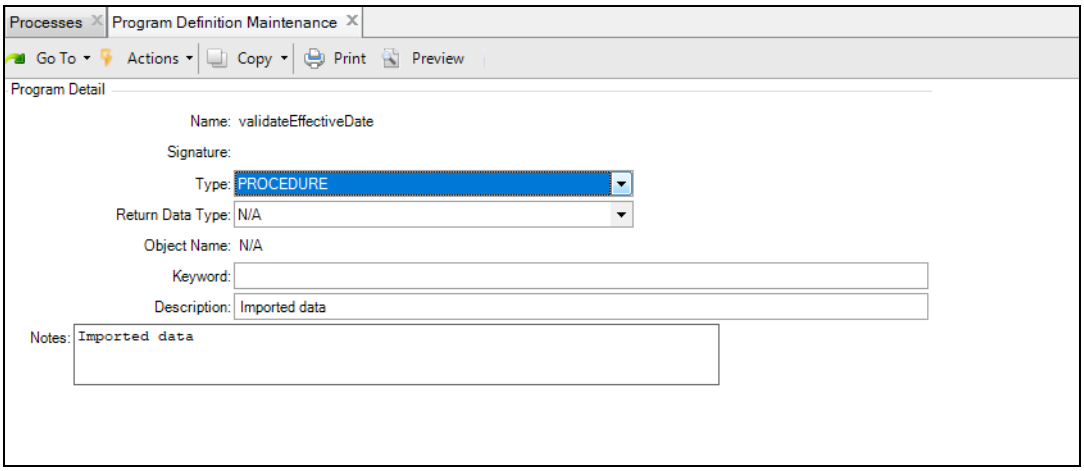
Edit Parameters. Enter Yes or No to edit program parameters. When you select Yes, the Parameter Details frame displays. When a record exists for Program ID, the system sets this field to Yes. For new program records, the default is No.

Note When the file is a class file, this field is not editable.

Notes. Enter any notes or comments about the program, procedure, or function.

Program Detail Frame

Fig. 2.8
Program Definition Maintenance



Name. Enter the name of the function or procedure in the program. The name must be unique for the Parent ID and cannot be blank.

Note If this is a class file, this field indicates the method.

Signature. Identifies the signature for the method that is listed in the Name field.

Note This field is part of the index for the program details because class files can have the same method name defined multiple times within the same file with different signatures. It consists of the parameter sequence number, a parameter type identifier, and the data type identifier for each parameter in the signature. When being displayed to the user in a readable format, each parameter data type is displayed, separated by a comma. When the parameter is a type of an output, then OUTPUT or INPUT-OUTPUT is displayed before to the data type.

This field is display only.

Type . Enter the program type. Choose from:

1: Function

2: Procedure

Return Data Type. Enter the data type for the value being returned by a function. Choose from:

- CHARACTER
- DATE
- DATETIME
- DATIME-TZ
- DECIMAL
- HANDLE
- ILIST*
- INTERGER
- INT64
- LOGIAL
- LONGCHAR
- MEMPTR
- OBJECT*
- RAW
- RECID
- ROWID

Note The IList and Object data types are only associated with class files.

- **Object.** A data object being passed to or from a method.
- **IList.** A list of messages that may be passed to or from the methods.

Object Name. When the data type is set to Object, this field displays the name of the object that is defined for the particular method that is listed in the Name field. This field is display only.

Keyword. Enter key words to describe this procedure or function so that you can easily find the data associated with the keyword—such as an item number—when you run reports

Description. Enter a short description to display on reports.

Notes. Enter any notes or comments about the program, procedure, or function.

Parameter Detail Frame

You enter the parameter type by selecting from a drop-down list of parameter types that determine where the program runs. Parameter entry is regulated by the value you enter for the data type. When you specify to edit parameters, the system displays the Parameter Details frame.

Fig. 2.9
Program Definition Maintenance

Sequence. Enter the sequence in which the system passes parameters to the program, function, or procedure.

The sequence must be unique and greater than 0 (zero) for the Parent ID, and greater than 0 for the program detail ID or procedure or function.

Name. Enter the parameter name that the system passes for the program, procedure, or function.

Original. Displays the “Original Name” of the parameter “Name.” This field is display only.

Note This value is set during the program import. If the “Name” is changed in the future, the original name can be viewed in order to give the user a better understanding of what the “Name” currently represents. This is helpful when viewing the event parameters.

Type. Enter the parameter type. Choose from:

- 1: Input
- 2: Output
- 3: Input-Output
- 4: Buffer

Datatype. Enter the parameter data type. Choose from:

- BLOB
- CHARACTER
- CLOB
- DATASET
- DATASET-HANDLE
- DATE
- DATETIME
- DATEIME-TZ
- DECIMAL
- INTEGER
- INT64
- LOGICAL
- LONGCHAR
- MEMPTR
- N/A
- RAW
- RECID
- ROWID
- TABLE-HANDLE
- WIDGET-HANDLE

This field works with the parameter you enter in the Name field.

Extents. Specify the number of extents for the parameter.

Table. Specify the name of the table that is passed as the parameter. When Data type is Table or the Parameter Type is a buffer, then this field cannot be blank.

Import/Export Programs

You use the programs in the Data Collection Utilities (6.23) menu to import/export data to and from the RF device, mobile phone, or other field device and the Data Collection engine that runs on the AppServer.

All Data Collection Import/Export programs present the same fields, except that Event Template and Lookup Template Import/Export do not have an Output File field. You can use the discussion on page 35 to follow the flow and understand fields in the import/export programs within Data Collection.

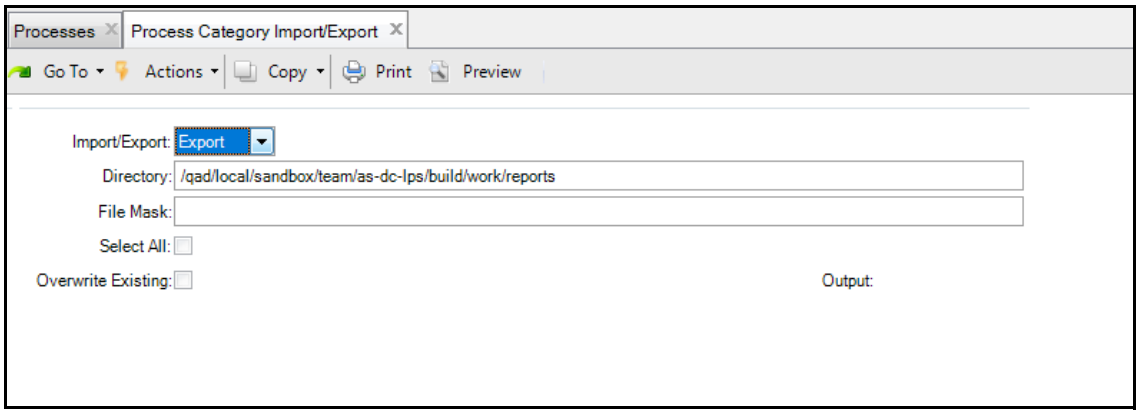
You can use the import/export programs described in the following topics to import/export, but for lookup and event templates, you can also import and export while you are creating a transaction event or lookup from within Transaction Definition Maintenance (6.16). The following topics discuss importing/exporting of lookup and event templates from within the character and .NET UI versions.

Process Category Import/Export

Process categories are used to group functionality for the mobile field device transactions. The categories define the menu structure for the RF device. Use Process Category Import/Export (6.23.1) to import/export process categories to and from the RF device, mobile phone, or other field device and the Data Collection engine that runs on the AppServer.

For more information about process categories, see “Create Process Categories” on page 90.

Fig. 2.10
Process Category Import/Export



Import/Export. Indicate whether you are importing or exporting files.

Directory. If you are importing files, enter the full path where the files exist, or if you are exporting, enter the full path where the files will be exported.

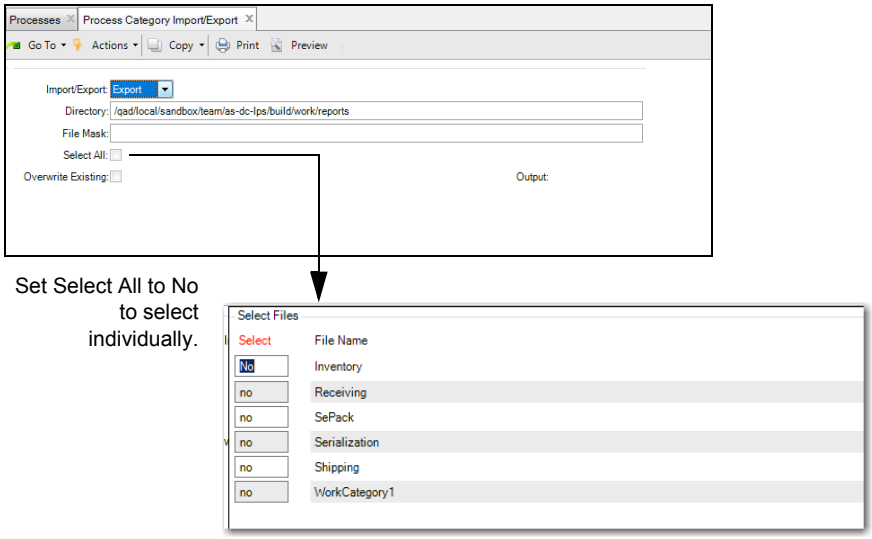
File Mask. Specify one or more patterns, including wildcards (*), for the system to use in selecting files for importing/exporting. For example, if you enter *.EDW, the system selects all files with an extension of EDW. Separate multiple entries with commas.

The system associates the values you enter with your user ID. Next time you run this program, the field defaults to the same values you entered previously.

Select All. Choose one of the following:

- **No**. A window displays letting you include and exclude selected files. Toggle between Yes and No in the Select field in the frame that displays to select the files to import/export.
- **Yes**. Automatically selects all files to import/export.

Fig. 2.11
Process Category Import/Export



Overwrite Existing. Specify Yes for the system to overwrite any existing file names in the application version you specified (environment).

Program Definition Import/Export

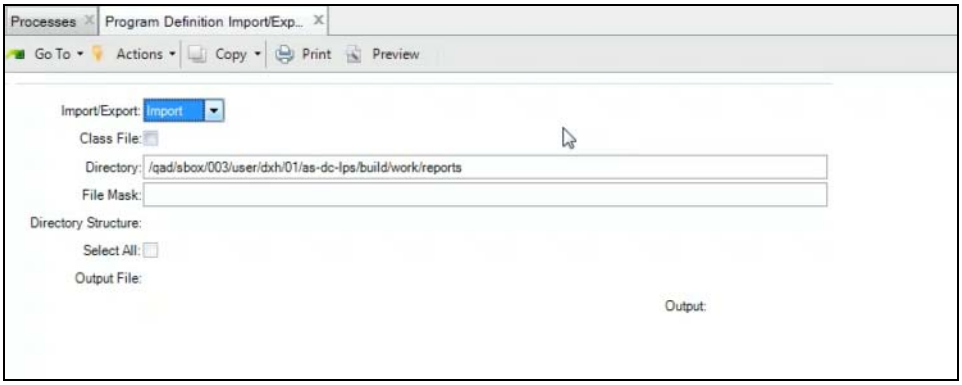
The system uses the program definitions to validate and execute predefined logic for transactions you develop to run on a mobile field device or to call an API for performing transactions using QAD's Data Collection module. Use Program Definition Import/Export (6.23.2) to perform the following tasks:

- Import/export program definitions to and from a specific application version (environment).
- Import/export class files.

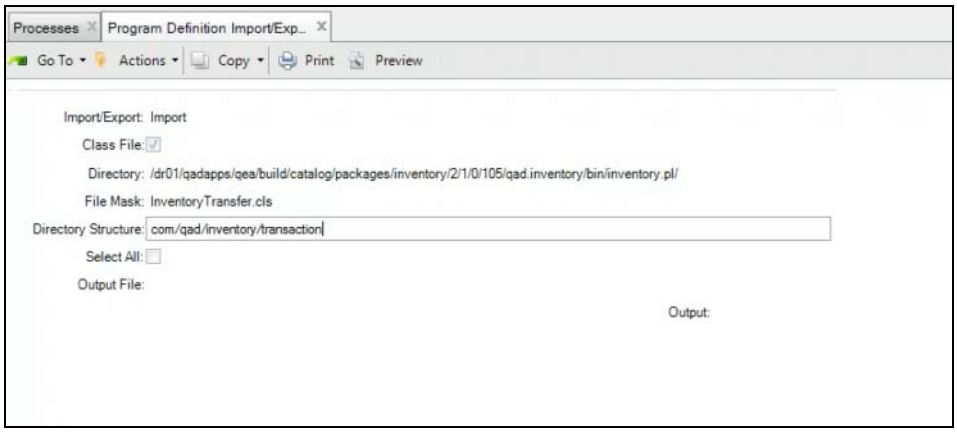
For more information about program definitions, see "Define the Program" on page 22.

Fig. 2.12
Program Definition Import/Export

Program Definition



Class File



Import/Export. Indicate whether you are importing or exporting files.

Note Only precompiled source code can be imported.

Class File. Select this check box if you are importing a class file. Only a single class file can be imported at one time, unlike 4GL code, where multiple files can be selected. For more information about class files, see the documentation provided by Progress.

Directory. If you are importing files, enter the full path where the files exist, or if you are exporting, enter the full path where the files will be exported.

File Mask. Specify one or more patterns, including wildcards (*), for the system to use in selecting files for importing/exporting. For example, if you enter *.EDW, the system selects all files with an extension of EDW. Separate multiple entries with commas.

The system associates the values you enter with your user ID. Next time you run this program, the field defaults to the same values you entered previously.

Note If you are importing a class file, you must enter the exact file name that you are importing; for example, `InventoryTransfer.cls`.

Directory Structure. Enter the path to the file on the disk that you want to run. When this field is blank, it is assumed that the directory structure is `us/<xx>`, where <*xx*> represents the first two characters of the file. For example, the directory structure for `sosomt.p` is `us/so`. In this example, this field would be left blank.

An example where this field would be populated is for `APIDispatcher.p` because the directory structure for this program is `com/qad/erp/api.` In this example, you would enter `com/qad/erp/api` in this field.

Note If you are importing a class file, this field cannot be blank and you must enter the directory structure of the class file; for example, `com/qad/inventory/transaction`.

Output File. Enter the name of the output file. This field is only used when exporting.

Select All. Choose one of the following:

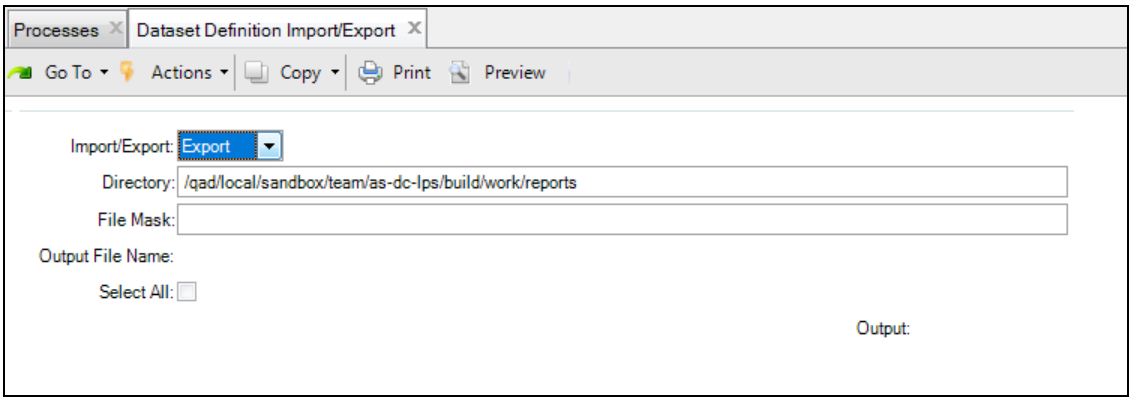
- **No**. A window displays letting you include and exclude selected files (see Figure 2.11). Toggle between Yes and No in the Select field in the frame that displays to select the files to import/export.
- **Yes**. Automatically selects all files to import/export.

Dataset Definition Import/Export

Use Dataset Definition Import/Export (6.23.3) to import/export dataset definitions to and from a specific application version (environment).

For information about setting up dataset definitions, see “Define the Dataset” on page 21.

Fig. 2.13
Dataset Definition Import/Export



Import/Export. Indicate whether you are importing or exporting files.

Directory. If you are importing files, enter the full path where the files exist, or if you are exporting, enter the full path where the files will be exported.

File Mask. Specify one or more patterns, including wildcards (*), for the system to use in selecting files for importing/exporting. For example, if you enter *.EDW, the system selects all files with an extension of EDW. Separate multiple entries with commas.

The system associates the values you enter with your user ID. Next time you run this program, the field defaults to the same values you entered previously.

Output File Name. Enter the name of the output file. This field is only used when exporting.

Select All. Choose one of the following:

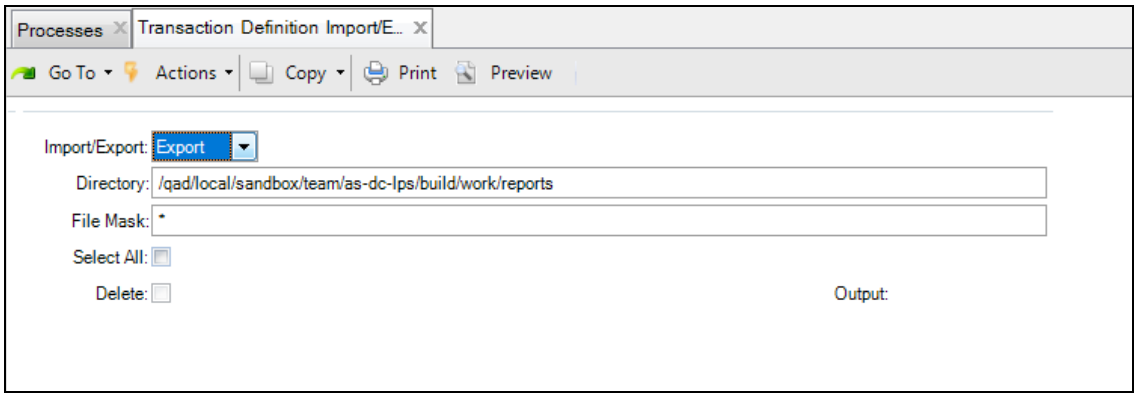
- **No**. A window displays letting you include and exclude selected files (see Figure 2.11). Toggle between Yes and No in the Select field in the frame that displays to select the files to import/export.
- **Yes**. Automatically selects all files to import/export.

Transaction Definition Import/Export

The transaction definition defines the transactions that run on the RF device and which business logic and events that are called during the transaction. Use Transaction Definition Import/Export (6.23.4) to import or export transaction definitions to and from a specific application version (environment).

For more information about transaction definitions, see “Create Transactions” on page 38.

Fig. 2.14
Transaction Definition Import/Export



Import/Export. Indicate whether you are importing or exporting files.

Directory. If you are importing files, enter the full path where the files exist, or if you are exporting, enter the full path where the files will be exported.

File Mask. Specify one or more patterns, including wildcards (*), for the system to use in selecting files for importing/exporting. For example, if you enter *.EDW, the system selects all files with an extension of EDW. Separate multiple entries with commas.

The system associates the values you enter with your user ID. Next time you run this program, the field defaults to the same values you entered previously.

Select All. Choose one of the following:

- **No**. A window displays letting you include and exclude selected files (see Figure 2.11). Toggle between Yes and No in the Select field in the frame that displays to select the files to import/export.
- **Yes**. Automatically selects all files to import/export.

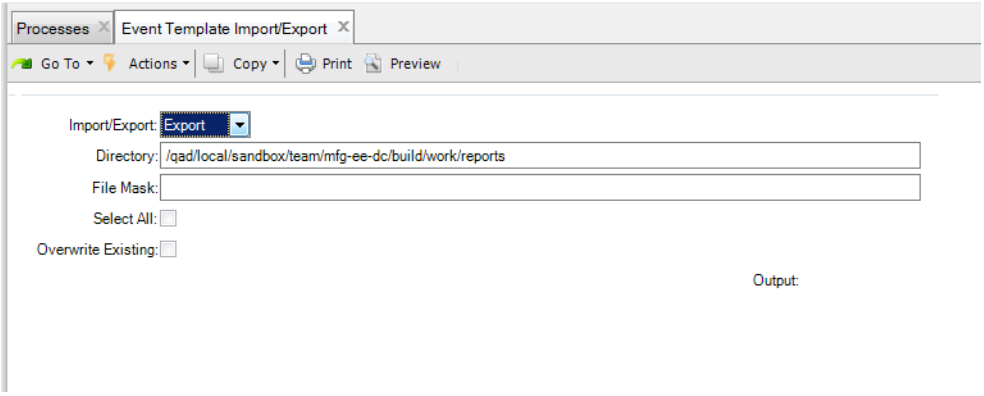
Delete. Select this check box to remove the transaction definition from the database after it is exported. This field is only used when exporting.

Event Template Import/Export

Event templates allow users to set up a templates for programs, functions, and so on, that the system calls as an event process in other transactions that you build. Use Event Template Import/Export (6.23.5) to import/export event templates to and from a specific application version (environment).

For more information about events and event templates, see “Create Events” on page 56.

Fig. 2.15
Event Template Import/Export



Import/Export. Indicate whether you are importing or exporting files.

Directory. If you are importing files, enter the full path where the files exist, or if you are exporting, enter the full path where the files will be exported.

File Mask. Specify one or more patterns, including wildcards (*), for the system to use in selecting files for importing/exporting. For example, if you enter *.EDW, the system selects all files with an extension of EDW. Separate multiple entries with commas.

The system associates the values you enter with your user ID. Next time you run this program, the field defaults to the same values you entered previously.

Select All. Choose one of the following:

- **No**. A window displays letting you include and exclude selected files (see Figure 2.11). Toggle between Yes and No in the Select field in the frame that displays to select the files to import/export.
- **Yes**. Automatically selects all files to import/export.

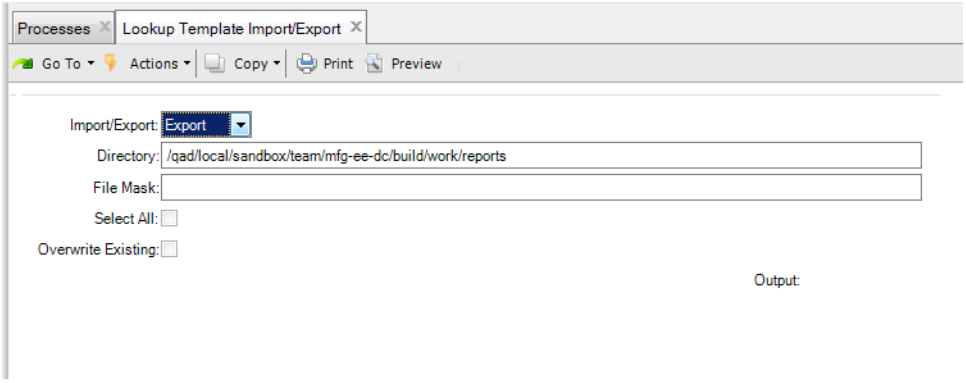
Overwrite Existing. Specify Yes for the system to overwrite any existing file names in the application version you specified (environment).

Lookup Template Import/Export

Use Lookup Template Import/Export (6.23.6) to import/export lookup templates to and from a specific application version (environment).

For more information about lookups and lookup templates, see “Create Lookups” on page 69.

Fig. 2.16
Lookup Template Import/Export



Import/Export. Indicate whether you are importing or exporting files.

Directory. If you are importing files, enter the full path where the files exist, or if you are exporting, enter the full path where the files will be exported.

File Mask. Specify one or more patterns, including wildcards (*), for the system to use in selecting files for importing/exporting. For example, if you enter *.EDW, the system selects all files with an extension of EDW. Separate multiple entries with commas.

The system associates the values you enter with your user ID. Next time you run this program, the field defaults to the same values you entered previously.

Select All. Choose one of the following:

- **No**. A window displays letting you include and exclude selected files (see Figure 2.11). Toggle between Yes and No in the Select field in the frame that displays to select the files to import/export.
- **Yes**. Automatically selects all files to import/export.

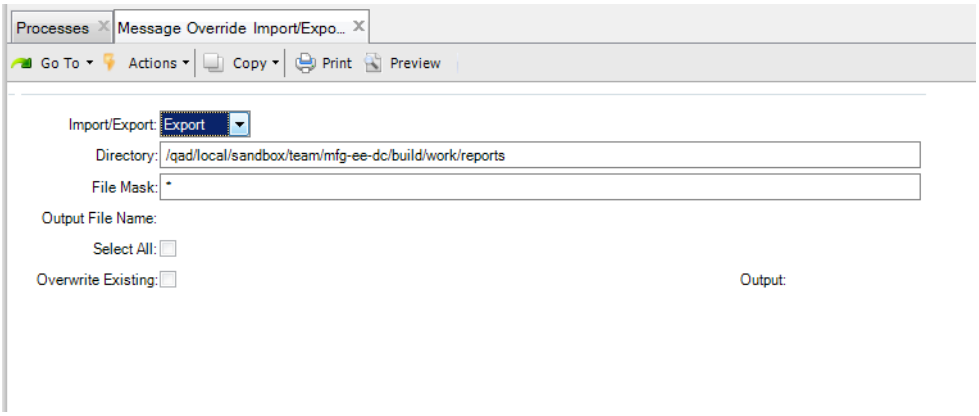
Overwrite Existing. Specify Yes for the system to overwrite any existing file names in the application version you specified (environment).

Message Override Import/Export

Message overrides specify how the system processes an application message number when it receives the message during a Data Collection transaction. Use Message Override Import/Export (6.23.7) to import/export message overrides to and from a specific application version (environment).

For more information about message overrides, see “Create Message Overrides” on page 74.

Fig. 2.17
Message Override Import/Export



Import/Export. Indicate whether you are importing or exporting files.

Directory. If you are importing files, enter the full path where the files exist, or if you are exporting, enter the full path where the files will be exported.

File Mask. Specify one or more patterns, including wildcards (*), for the system to use in selecting files for importing/exporting. For example, if you enter *.EDW, the system selects all files with an extension of EDW. Separate multiple entries with commas.

The system associates the values you enter with your user ID. Next time you run this program, the field defaults to the same values you entered previously.

Output File Name. Enter the name of the output file. This field is only used when exporting.

Select All. Choose one of the following:

- **No**. A window displays letting you include and exclude selected files (see Figure 2.11). Toggle between Yes and No in the Select field in the frame that displays to select the files to import/export.
- **Yes**. Automatically selects all files to import/export.

Overwrite Existing. Specify Yes for the system to overwrite any existing file names in the application version you specified (environment).

Importing/Exporting Lookup/Event Templates from Character EE

You can import lookup and event template data in the character version when you run Transaction Definition Maintenance (6.16). To do this, press ESC-I (the letter i) from the Lookup or Event Frame. The system displays a list of templates for you to select.

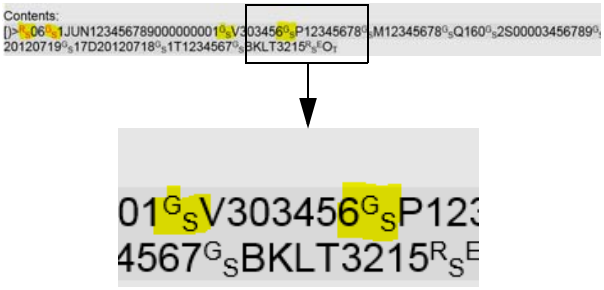
You can export lookup and event template data in the character version when you run Transaction Definition Maintenance (6.16). To do this, press ESC-E from the Lookup or Event Frame. The system prompts for the template name; then, exports the template.

Importing from Lookup/Event Templates from .NET UI EE

When you run Transaction Definition Maintenance from .NET UI, you can use the .NET UI Action button to import and export data in the Event or Lookup Frame. To do this, click Action, then select either Import Data or Export Data. The system provides a frame for you to enter details of the data you want to export, such as frame name, field name, and so on. Refer to your .NET UI user documentation for more information on importing/exporting within .NET UI.

Troubleshooting

The following table provides troubleshooting tips.

Problem	Solution
1. The Admin changes a user's password. 2. The user logs into .NET and is prompted to change their password. 3. After changing their password, the user opens a collection and attempts to use a tab running a DC transaction. 4. The user is prompted for their password. 5. The user enters the new password but receives an invalid password message.	When you change your password, you must log out of .NET and then log back in *before* trying to use DC within .NET.
Data Collection doesn't support the special characters marked in yellow: 	This issue has been fixed in Data Collection v3.1 Install Data Collection 3.1 (or later).
There is no ability to view what data is passed to the API. Older versions of Data Collection do not have the functionality to print the data (configuration data, api data) to file. This data can be used for troubleshooting purpose.	Install Data Collection v3.0 (or later). Data Collection v3.0 introduced a new program called dctransprocxr.p, which has the following procedures that can print data to file to be used for troubleshooting: • WriteOutConfiguration for config-dataset (for configuration data) • WriteAPIdataset for API-DATASET (for data passed to API) • WriteUIdataset for UI-CONFIG-DATASET (for User interface configuration data)

Chapter 3

Create Transactions, Events, Lookups, and Message Overrides

This chapter discusses the following topics:

Overview

Once you set up Data Collection, you are ready to create transactions based on the dataset and program definition you imported and modified or created for the environment in which you want the transaction to be based.

You build the transactions, which include building events of all kinds, using Transaction Definition Maintenance (6.16). Building the transactions is the center of Data Collection functionality. This is because you build the transaction, including pre-process, post-process or commit events attached to tables, lookups, fields; build the lookups; edit parameters and more. You are building the core of the mobile field device's menu transactions.

When you set up the transactions and events, you can optionally save the following to templates that you originally created while creating the transaction:

- Events or sequence of events to an event template
- Lookups to a lookup template

When you create other transactions that call the same events, you can use function keys to select the event or lookup template name while you are in Transaction Definition Maintenance to save time.

Note Due to mobile field device size constraints, you cannot use the structure QAD EE field lookups, so you either create or import previously created lookups into the transactions.

Create Transactions

Use Transaction Definition Maintenance (6. 16) to define transactions that you want to run on the RF device. Use the program to determine what business logic and events you want to call during the transaction. You can call the QAD EE standard programs and code or customized programs and code directly.

You import a dataset definition into the system and configure the buffer fields for user input. When creating a transaction definition, the buffer names and their associated fields are imported into the definition. You can then reorder the sequence in which the fields for the buffer display. You can also insert user-defined fields into a buffer, which can then be used as local variables for calculations, informational displays to the user, user prompts for additional information to populate the dataset, and so on.

You also use this program to set up the events and the timing of events. For example, you can determine whether an in-line validation occurs, such as verifying whether an item exists before committing the transaction or having custom default locations display while processing. You also determine when you want pre-processing, post-processing, or commit events to occur. When maintaining an event, you can add a message number that can be displayed when the event fails.

You can also determine whether transactions are committed to the database as a unit or on a line-by-line basis or based on an event, such as committing after every quantity decrease.

Note the following:

- When defining a lookup event, you cannot access the parameter section.
- When a Transaction definition is deleted, all records linked to the transaction must be deleted.

Navigation

This program consists of many frames that let you drill down into the setup of the transaction, including transaction details, pre-and post-process event data, buffers, fields, parameters, and other details. Once you enter a new frame, you typically enter data in the top fields, then press Enter, which enables other fields in the frame.

Note The general flow of transaction setup for most of the program follows the same flow once you pass the Transaction Definition header frame. You drill down, entering data, in this flow maintaining:

1. Datasets (highest level)
2. Buffers (tables)
3. Fields
4. Lookups

When you create/edit each of these levels, the process is typically the same:

1. Start by entering a sequence and description.
2. Indicate if the item is active or has details.
3. Go on to enter detailed information.

Transaction Maintenance Flow

The transaction maintenance flow—presented through a series of frames—that you encounter when you choose to edit all data for every portion of the transaction is as follows:

- “Set Up Transaction-Identifying Data” on page 40

  In the program header, enter transaction-identifying data in the header frame, then, optionally go on to edit the application version.
- “Set Up Transaction Definition UI Style and Processing Details” on page 42

  Enter the UI style and processing details. Specify the imported dataset (XSD) information, enter Response XSD data, specify whether you want to create events and the type, and specify whether to edit the dataset profile, pre-/post-process, or commit events, and more.
- “Edit Dataset Profile” on page 45

  Edit the dataset directly in the frame that displays when you set the Edit Dataset field.
- “Maintain Dataset Buffer Details” on page 46

  Edit dataset buffer details, by selecting records from a list of API dataset records for the UI configuration.
- “Maintain Dataset Buffer Field Information” on page 49

  When you set the Fields field to Yes when viewing transactions, you can go on to enter field information.
- “Maintain Field Details” on page 51

  Enter more details on the field, including whether the field has default, validation, pre-/post-processing events.

- Maintain processing events and event parameters; see “Create Events” on page 56

  Important You can maintain events at the dataset, buffer (table), or field/lookup level, which means that the frames to create/maintain the event can be interjected within the transaction flow at the level at which you choose to create the events.
- Maintain field lookups; see “Create Lookups” on page 69
- “Edit Parameters” on page 54

  The system presents a frame to edit parameters.

Set Up Transaction-Identifying Data

Use the header frame in the program to identify the transaction as follows:

1. Specify the transaction name and transaction version.
2. Click Yes to debug the transaction and specify a debug level; then, press Enter to edit other data in the frame.

   When creating a transaction definition, the system prompts to enter the transaction setup history. This is for tracking purposes and does not affect transaction processing. Go on to step 3; otherwise, go on to step 4.
3. Enter the unique identification code that defines the functional flow for this transaction (that is, JIRA tickets, SUI tickets, and so on).

   The system maintains a transaction history information log with the ECO, modification date time stamp, user ID of user who entered the history log, transaction version, and related notes.
4. Specify the transaction description, a defined application name, menu name, and menu program.

   You enter the application information not only to identify this transaction but also for security. To build the transaction, you must have access to the QAD EE application menu and program.

   Note The Child Transactions field displays the number of child or linked transactions the system calls from this transaction definition. You cannot edit this field.
5. Choose one of the following:

   a. Specify Yes to enter notes related to this transaction definition.

      The system displays a frame to add notes. Existing notes display in the frame; you can edit the notes.

   b. Press Enter to skip the notes. Go on to the next step.
6. Choose one of the following:

   a. Specify Yes to enter additional transaction history information.

      The system displays a history log with any existing transaction history. The log shows the ECO, modification date time stamp, user ID of user who entered the history log, transaction version, and related notes. You can scroll through existing logs and edit the content or enter new history information.

   b. Press Enter to not enter history information, then go on to the next step.

7 Choose one of the following:

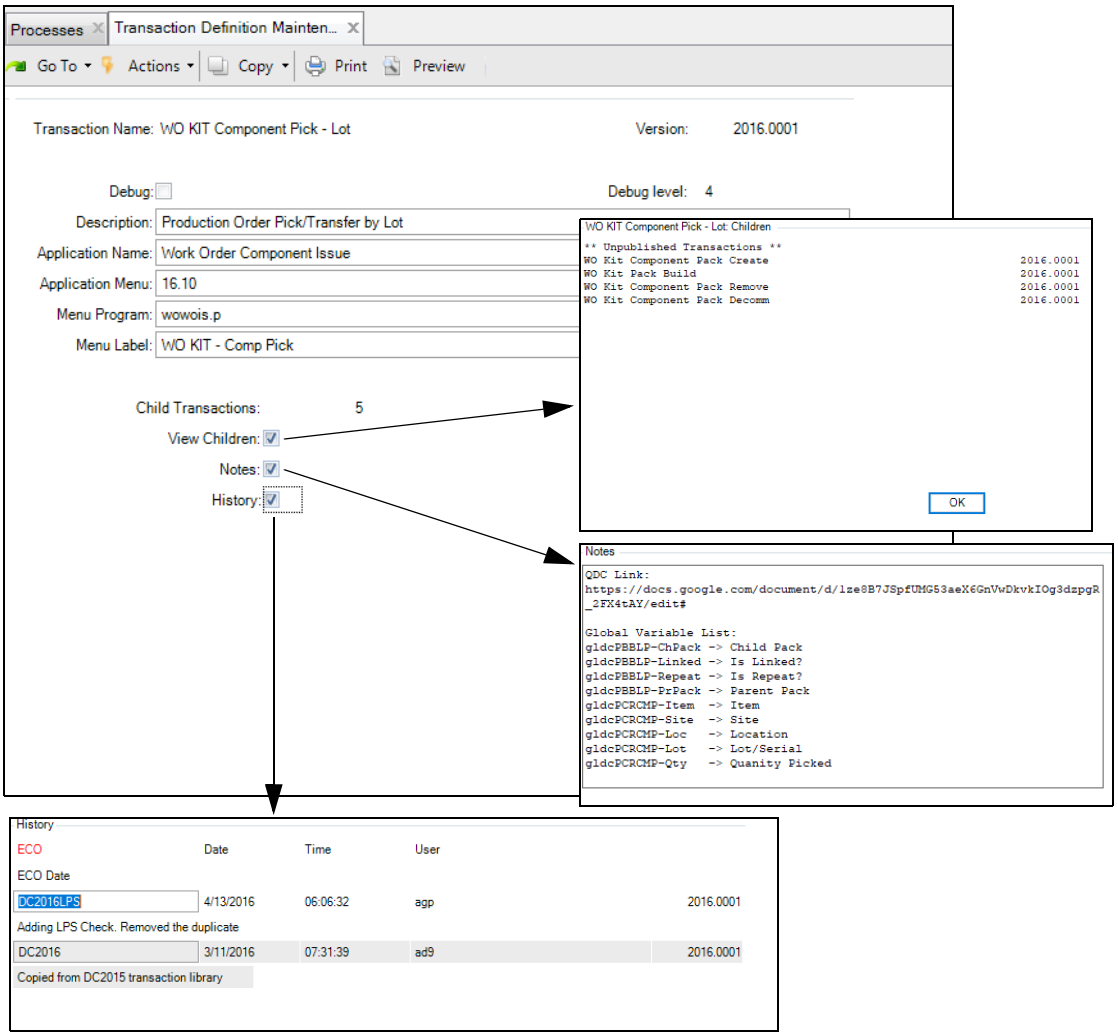
a Specify Yes to edit the application version.

The system displays the Application Version frame and lets you enter the application version and description; for example, for QAD EE version 2012.1, enter 2012.1.

b Press Enter to not edit the application version, then go on to set up the transaction definition UI; see “Set Up Transaction Definition UI Style and Processing Details” on page 42.

In the following topics, procedural and field information is associated with the various frames that display, depending upon your edit responses.

Fig. 3.1
Transaction Definition Maintenance, Header



Transaction Name. Enter the name of the transaction to run on the RF device. It is best to use meaningful names that represent the actual transaction; for example, enter Cycle Count Storage 2.

Version. Assign a version to the transaction.

Debug. Specify Yes to debug the transaction. When Yes, you can debug messages in the Data Collection engine and additional information. When Yes, the system creates a debugging log file, enables 4GL trace, and you can enter the Debug Level.

Debug Level. Enter the level of debugging. This field is display only unless you set Debug to Yes.

Description. Enter a description of the transaction.

Application Name. Enter the application name associated with the transaction, such as EAM, TMS, and so on.

Application Menu. Enter the application menu that the transaction uses. For example, enter 5.13.1 for a PO receipt.

You can now specify a menu label. When you add a task detail menu item to the menu, the system defaults the menu label you specify in the transaction definition as the menu label. When you modify an existing task detail menu item, the menu label does not change.

Menu Program. Enter the name of the QAD EE program for which you entered the menu number. For example, enter poporc.p when you enter 5.13.1 as the Application Menu number.

Menu Label. Enter the label for this menu program.

Child Transactions. Indicates the number of child transactions that are referenced by the parent transaction.

View Children. Select this check box to view the child transactions that are referenced by the parent transaction.

Notes. Select this check box to view any notes for the transaction.

History. Select this check box to view the Transaction History Information.

Set Up Transaction Definition UI Style and Processing Details

In the Transaction Definition Frame, enter data as follows:

1. Enter the UI style.
2. Enter the time in seconds for processing timeouts and wait time.
3. Specify Yes or No to auto-commit and auto-repeat the transaction.

   Auto-commit lets the user commit without additional prompting and data entry. Auto-repeat lets the user enter additional data without reprompting; for example, the user can enter several different POs for a PO receipt transaction without committing the receipt or additional system prompting. When Auto Repeat is disabled, the auto-repeat is non-functional.
4. Specify Yes or No to delete failed transactions.

   When you set Delete Failed Trans field to Yes, the system retains the transaction data that users enter if there is a transaction failure so that the user can make corrections.

   The system defaults pre-process, post-process, or commit events as existing or not existing in the dataset. These fields are for information only.
5. Enter the dataset (XSD) name and whether you want to commit the dataset.

6 Enter the XSD response data.

7 Enter a valid QXtend Controller's XML API name.

8 Indicate if you want to edit the dataset profile.

When you finish in the current frame, the system displays the dataset profile in a separate frame for you to edit. You can edit within the frame and save your changes.

9 Indicate whether you want to edit events that exist for the dataset (pre-/post-/commit events).

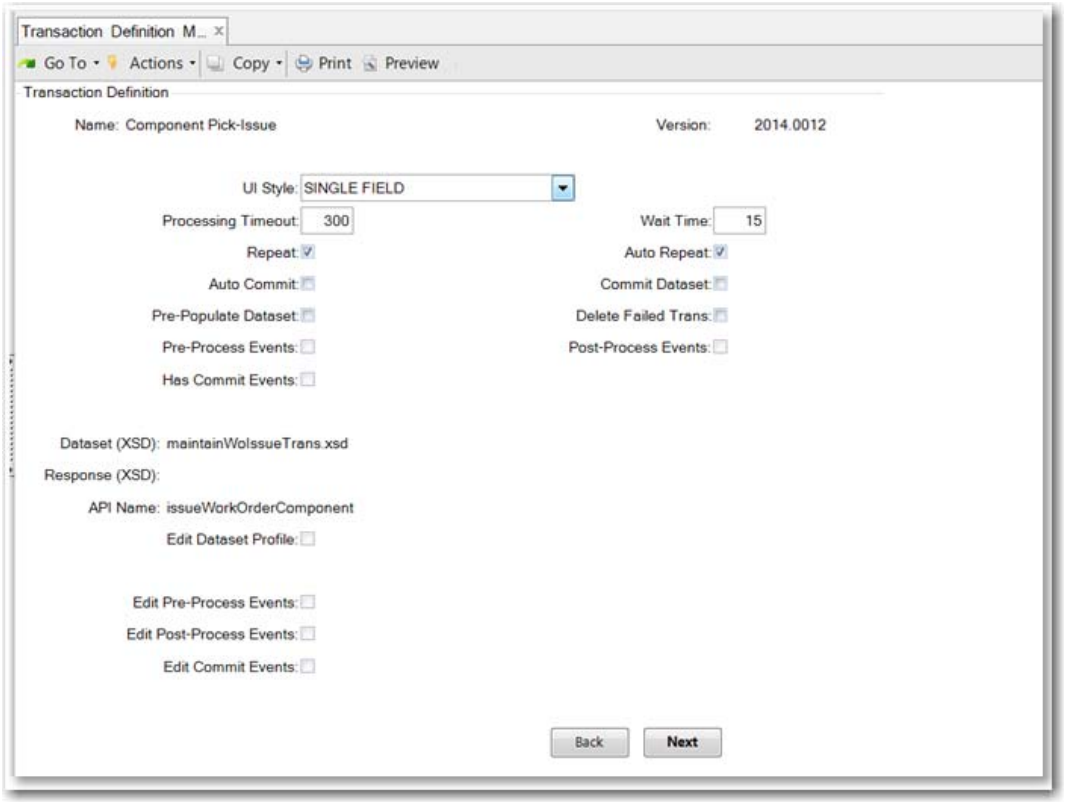
10 Press Go.

The system populates the transaction with data from the datasets. The system goes on to display additional frames, depending upon what you set to edit. When you do not edit the dataset profile, you can go on to edit the events associated with the dataset.

You can press Enter and bypass the Event editing, to see the XSD buffer definitions that are in the dataset profile; see "Maintain Dataset Buffer Details" on page 46.

11 When you are ready to save your dataset, you can commit at the highest dataset level by setting the Commit Dataset field to Yes in the UI Style frame.

Fig. 3.2
Transaction Definition Maintenance, UI Style



UI Style. Specify the RF UI style. The value defaults from Data Collection Control (6.22.24). Choose from:

- **Multiple Fields.** Multiple fields per mobile field device screen, up to ten fields per screen so that the user can tab through the fields.

- **Single Fields.** Single field per mobile field device screen to scan data in.

Processing Timeout. Specify the time in seconds before the RF transaction completes. This is used for record locking. When the processing time elapses, the system notifies the user that the processing timed out, so records may be locked. The value defaults from Data Collection Control (6.22.24).

Wait Time. Specify the time in seconds for the user to wait before the system notifies the user that there is a record lock. The value defaults from Data Collection Control (6.22.24).

Repeat. Specify Yes for datasets to repeat. The Auto Repeat function works in combination with the Repeat flag. When Repeat is No, only one transaction can take place at a time.

Auto Repeat. Specify Yes to automatically start a new transaction. For example, once a PO is committed for a PO receipt, the user can simply enter another PO without the system prompting to start another PO receipt transaction. This field defaults from Data Collection Control (6.22.24).

Auto Commit. Specify one of the following:

- **No.** The system prompts the user to commit the transaction.
- **Yes**. The system automatically commits the transaction without prompting the user to commit. For example, the system can automatically commit after a complete PO receipt. The system sends the data that it collects thus far from the transaction to the API so that QAD EE can receive the transaction data and create the transaction in QAD EE.

Commit Dataset. Specify Yes to commit dataset changes.

Pre-Populate Dataset. Specify Yes to pre-populate the dataset.

Delete Failed Trans . Specify Yes to delete the entered transaction data in case a transaction fails. When No, the system preserves the last entered data even when the transaction fails.

Pre-Process/Post-Process/Commit Events. This is informational only, notifying you that there are events at the dataset level.

Dataset (XSD). Enter the name of a valid dataset (XSD) for this transaction as recognized in Dataset Definition Maintenance; see“Define the Dataset” on page 21.

Response XSD. Enter a valid response XSD to use with this transaction.

API Name. Enter a valid API method name (case sensitive) from the QAD QXtend Controller’s XML. Data Collection only uses the Controller’s XML. Refer to your QXtend documentation for more information on the Controller’s API.

Edit Dataset Profile. Enter Yes to edit the dataset profile. When Yes, the system displays an additional frame for you to edit the dataset definition; see “Edit Dataset Profile” on page 45.

Edit Pre-Process. Indicate whether you want to edit pre-process events. Pre-process events are those that are first presented to the Data Collection engine; for example, events that cause data to be pre-populated or records that can be used at the beginning of the transaction.

- **Yes.** The system displays additional frames to enter the sequence number and description for the pre-process event and specify whether the transaction is active or has details; see “Seq” on page 50.

- **No.** You cannot edit the pre-process data. Press Enter to continue. The system displays a frame to assign field names to a sequence and a short label and indicate Yes to update, display, or edit details; see “Edit Dataset Profile” on page 45.

Edit Post-Process. Specify one of the following:

- **Yes.** Enter Yes when the dataset has post-process events that you want to edit. Post-process events occur after the commit of the API transaction occurs; so, these events are typically cleanup tasks.

  The system displays a frame to enter the sequence number and description for the post-process event and specify whether the transaction is active or has details; see “Seq” on page 50.
- **No.** You cannot edit the post-process data. Press Enter to continue. The system displays a frame to assign field names to a sequence and a short label and indicate Yes to update, display, or edit details; see “Edit Dataset Profile” on page 45.

Edit Commit. Specify one of the following:

- **Yes.** Enter Yes when the dataset has commit events that you want to edit. Commit events are those that need to occur during the commit process.

  The system displays a frame to enter the sequence number and description of the commit transaction and indicate whether the Commit Transaction is Active or has details. When Yes, you follow the flow for editing pre- or post-process events; see “Seq” on page 50.
- **No.** You cannot edit the pre-process data. Press Enter to continue. The system displays a frame to assign field names to a sequence and a short label and indicate Yes to update, display, or edit details; see “Edit Dataset Profile” on page 45.

Edit Dataset Profile

When you set Edit Dataset Profile to Yes in the UI Style frame, the system displays a frame with the dataset profile content. You can edit directly in the screen that displays. You can make changes, then press Go to set the changes.

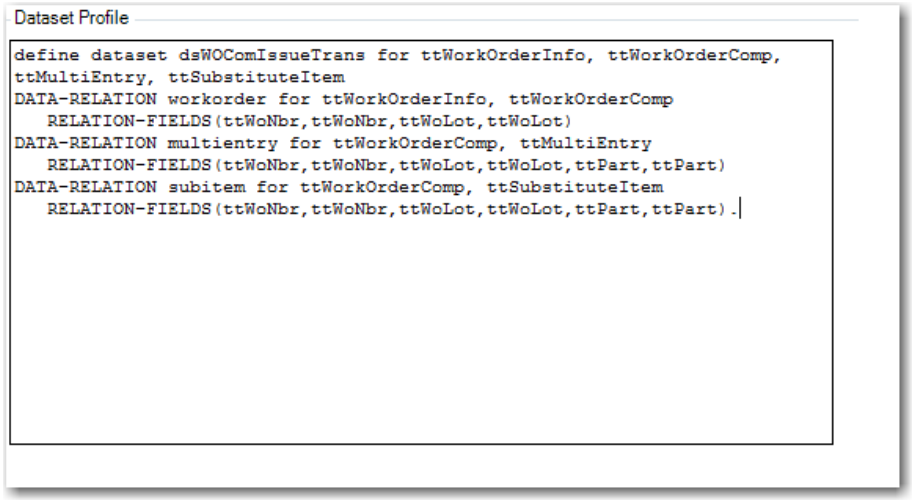
Important The changes are not committed until the system prompts you to confirm updating the dataset profile and you respond with Yes.

If you make a mistake, and you want to redisplay the original dataset profile for the API:

1. Remove all text from the screen so that the edit screen is blank.
2. Return to the header frame of Transaction Definition Maintenance.
3. Scroll down until you reach the Edit Dataset Profile field.
4. Set Edit Dataset Profile to Yes. (The field’s default is No.)
5. Press Go until you reach the screen that displays the dataset content.

   The original dataset profile content displays.

Fig. 3.3
Transaction Definition Maintenance, Edit Dataset Profile



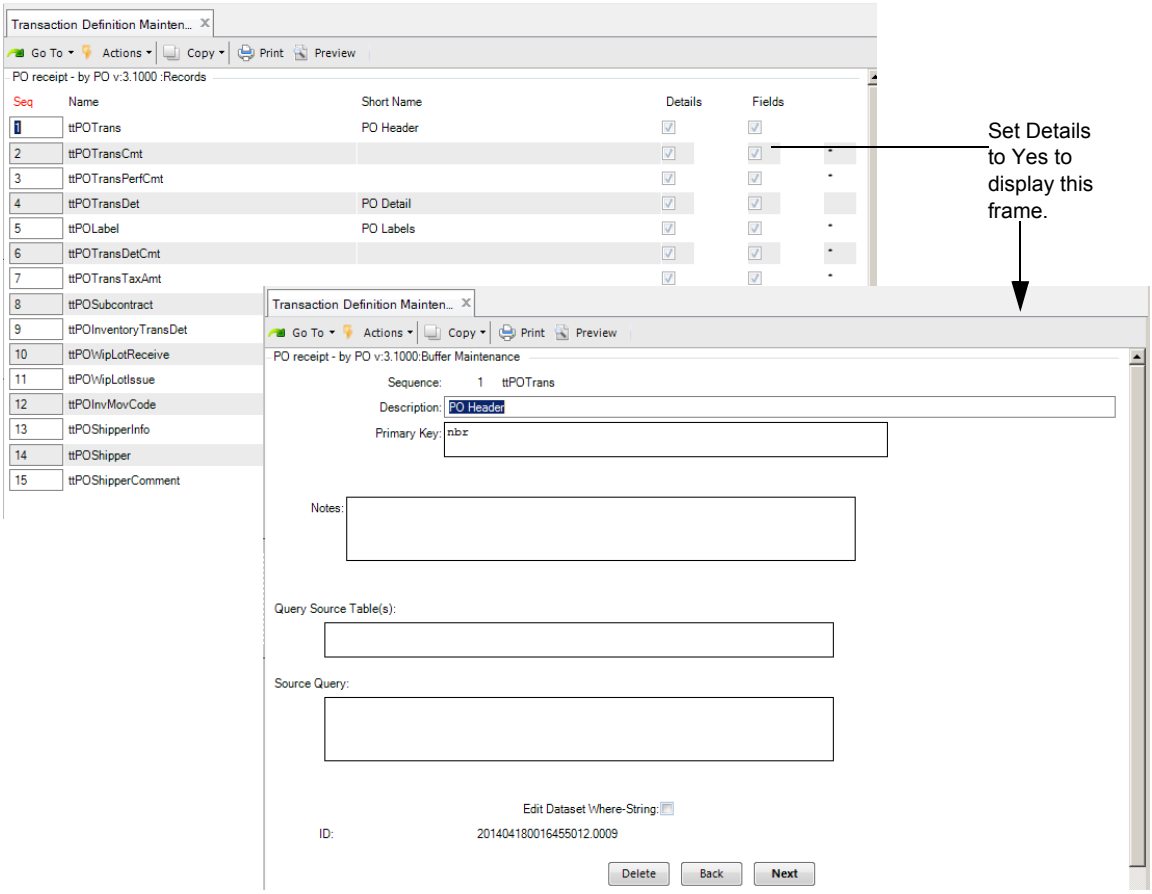
Maintain Dataset Buffer Details

You can edit the buffer (table) data that is included in the API dataset. You can do this after you set up the UI style by pressing Enter (to bypass the event editing.)

The system displays a frame that includes a list of the records that are part of the API dataset. You can peruse the list and determine which records are for the API dataset and which are for the UI configuration. Your goal is to collect data for transfer to the API.

In the screen, you can enter a short name for the buffer; then, indicate if you want to edit buffer details. Press Go to display the details.

Fig. 3.4
Buffer Data, Transactions and Details



Sequence. Enter the buffer sequence.

Name. Enter the name of the buffer. These are part of the API dataset, so you typically do not want to edit the name. The name indicates when the buffer is part of the dataset or part of the UI.

Short Name. Enter a short name for the buffer that the system displays to the mobile device user to give the user an idea of where the user is within the transaction.

Details. Indicate Yes to display and edit details (parameters and more) of the buffer definition of the dataset profile; see “Maintain Dataset Buffer Details” on page 46.

Fields. Enter Yes to edit fields.

Description. Enter a short description to describe the purpose of this buffer record.

Primary Key. When the system brings in the dataset definition that you specify, it brings in the keys for the buffer. You can change the primary key here.

Notes. Optionally, enter any notes about the record.

Query Source Table. Enter a valid QAD database table name. This field works in combination with the Source Query field and can be used to pre-populate buffer data.

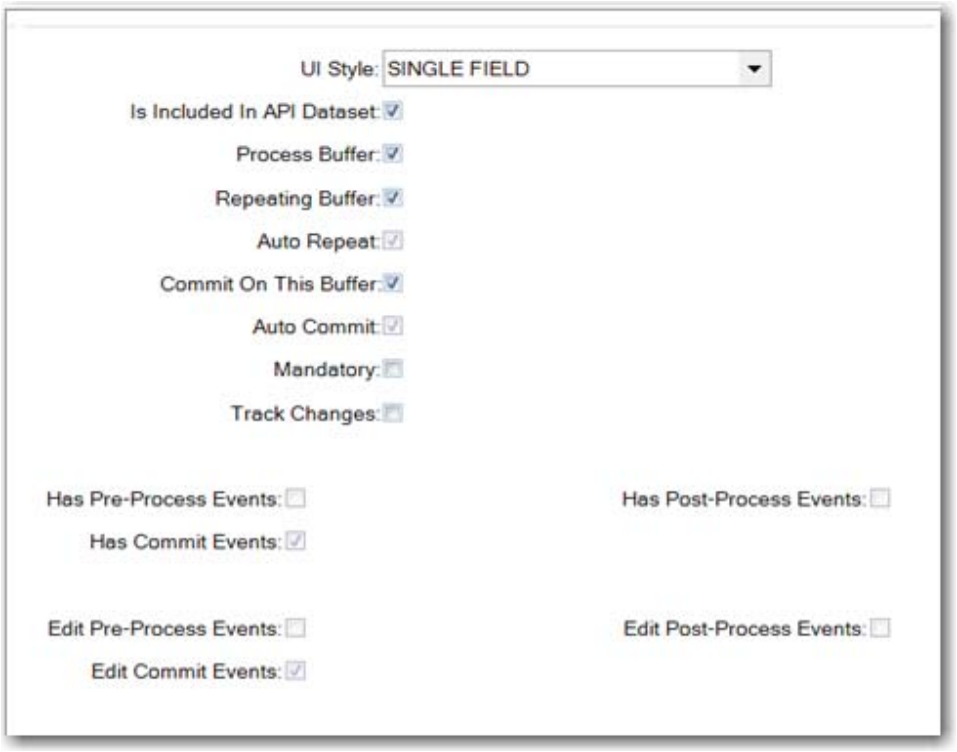
Source Query. Enter a valid Progress query string based on the Query Source Table. The Data Collection engine queries the database table to get a set of database records that the system uses to pre-populate the buffer.

Edit Dataset Where String. This field is used to specify a "where" string. It lets you enter criteria for what is allowed in this record; for example, it can be used in a query or pre-fetch data area to get POs that are available for a specific site or customer. It restricts the data available in the dataset.

Buffer Details UI Style

When you press Go after you set up the initial buffer maintenance data, the system displays a frame to let you enter the UI style.

Fig. 3.5
Buffer Data, UI Style



UI Style. This field is not implemented. You can change the UI style in this frame, but it has no effect on the transaction.

Is Included in the API Dataset. When No, an asterisk displays in the list of buffers to indicate that the buffer is not a part of the API dataset that you use. When you add a buffer, you must set this field to No; otherwise, the buffer's data is a part of the transaction data commits and it causes errors because the API cannot identify it.

Process Buffer. Indicate Yes or No during the transaction development to dynamically enable/disable record processing for the buffer (table):

- **Yes.** You are prompted for buffer data. The system processes all events and fields included in this buffer by the Data Collection engine during the transaction.

- **No.** You are not prompted for buffer data within this transaction. The engine skips this buffer during transaction processing.

Repeating Buffer. Indicate Yes if the buffer repeats. When Yes, the buffer repeats before it leaves the transaction; for example for a PO receipt, set to Yes so that users can repeatedly enter multiple PO for a single shipper receipt that can be sent as a single transaction.

Auto Repeat. Indicate Yes to automatically repeat the buffer without system interaction or prompting to do another transaction.

Commit on This Buffer. Indicate Yes to commit after the buffer. For example, for a receipt, you want to gather all items for the shipper and commit all items for a single receipt at one time.

Auto Commit. Indicate Yes to automatically commit without system interaction or prompting.

Mandatory. Indicate Yes to make this buffer mandatory. At least one record for this buffer must be entered.

Track Changes. Indicate Yes to enable change tracking for this buffer.

Has Pre-Process Events/Post-Process/Commit Events. Specify Yes if this buffer already has pre-/post-/commit events; see “Seq” on page 50.

Edit Pre-Process Events/Post-Process/Commit Events. Specify Yes to edit the buffer pre-/post-/commit events; see “Seq” on page 50.

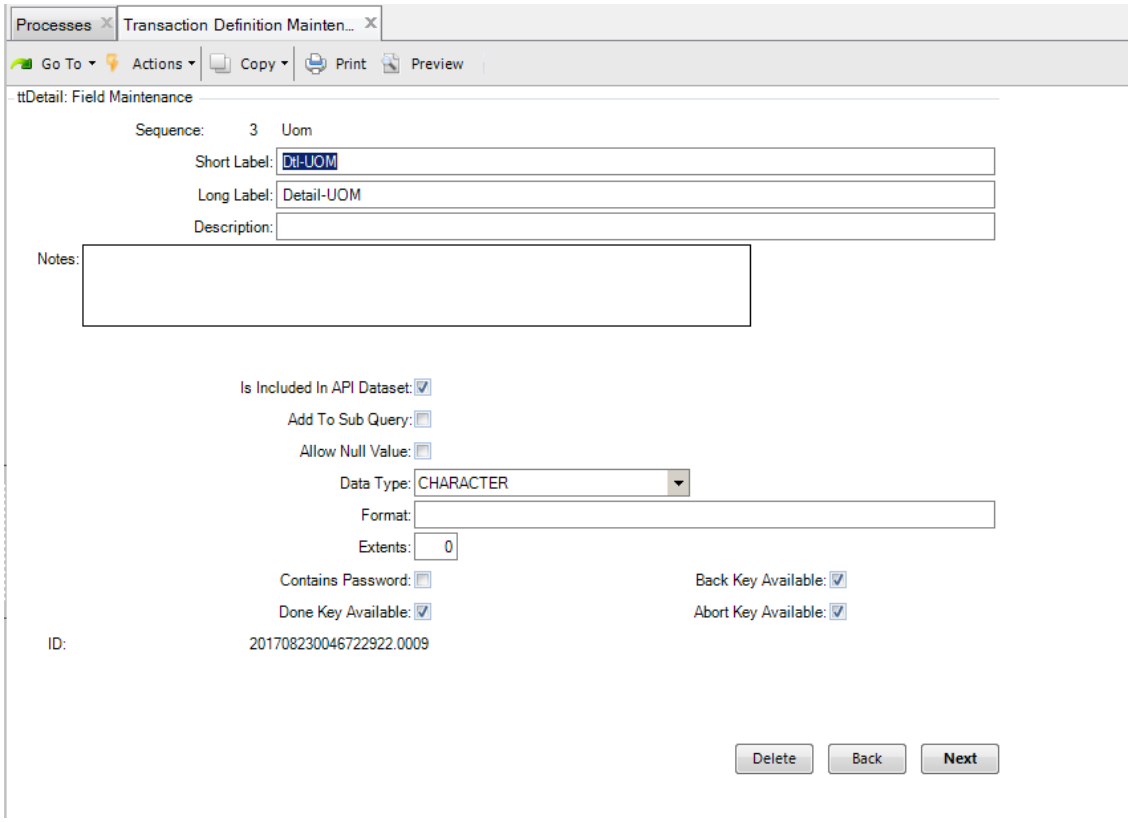
Maintain Dataset Buffer Field Information

Once you set up dataset buffer details, you press Enter to edit dataset buffer field information if you set Fields to Yes in the transaction screen; see Figure 3.4 on page 47.

You can assign fields to a sequence number, then specify the short label for the field. The short label displays on some RF screens during the transaction.

Once you set up the sequence numbers, the system displays a frame to let you enter additional field data, including long labels, descriptions, datatype, and extents, and to indicate whether the field is part of the API dataset or should be added to the subquery.

Fig. 3.6
Dataset Buffer Field Information



Seq. Enter a sequence number for the field to display on the UI.

Field Name. Enter the field name to display on the UI.

Short Label. Enter a short label for the field.

Update. Indicate Yes when the field data can be updated in the UI.

Display. Indicate Yes to indicate that the field is to be displayed in the UI.

Details. Indicate Yes to display details; see Figure 3.8 on page 52

Long Label. Enter the long label that can be displayed on the PC UI and single-line update for the mobile device.

Description. Enter a description of the field.

Notes. Optionally, enter any notes about the field.

Is included in the API Dataset. Specify Yes if this is a user-defined field or a dataset field.

Add to Sub Query. Indicate whether the system uses this field as one of the key fields for its child buffers.

The system uses the indication to pre-populate records into child buffers. So, when you set the field to Yes, the query string of a child buffer can use this field's value to retrieve or pre-populate records.

Allow Null Value. Indicate Yes to indicate if this field can have a null value as a valid value.

Data Type. Enter the datatype for the field.

Format. Enter the valid format of the field data based on the data type of the field.

Extents. Enter the number of extents for the field. If the field has one or more extents, the system displays the field extent frame (see Figure 3.7). You should enter the sequence number for every extent to configure its functionality. The field extent has specific events that you can configure.

Contains Password. If this field is set to Yes, the data that is entered in this field will not be visible to the user during the transaction execution. By default, this field is set to No.

Back Key Available. When the user is prompted to enter data into the field, specify whether the Back key will be displayed and available to the user. By default, this field is set to Yes.

Note During the processing of the transaction, this setting can be changed through the use of events.

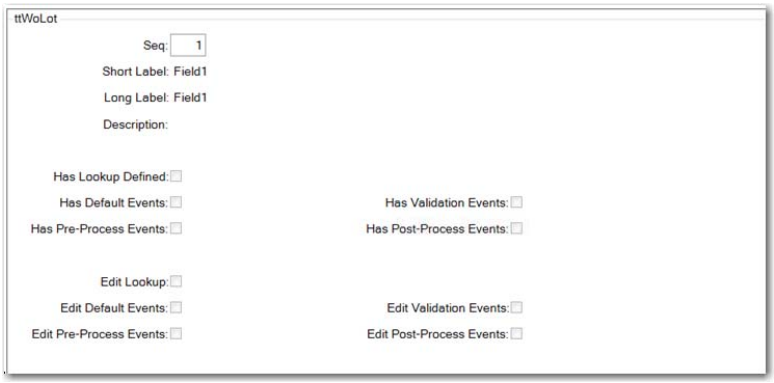
Done Key Available. When the user is prompted to enter data into the field, specify whether the Done key will be displayed and available to the user. By default, this field is set to Yes.

Note During the processing of the transaction, this setting can be changed through the use of events.

Abort Key Available. When the user is prompted to enter data into the field, specify whether the Abort key will be displayed and available to the user. By default, this field is set to Yes.

Note During the processing of the transaction, this setting can be changed through the use of events.

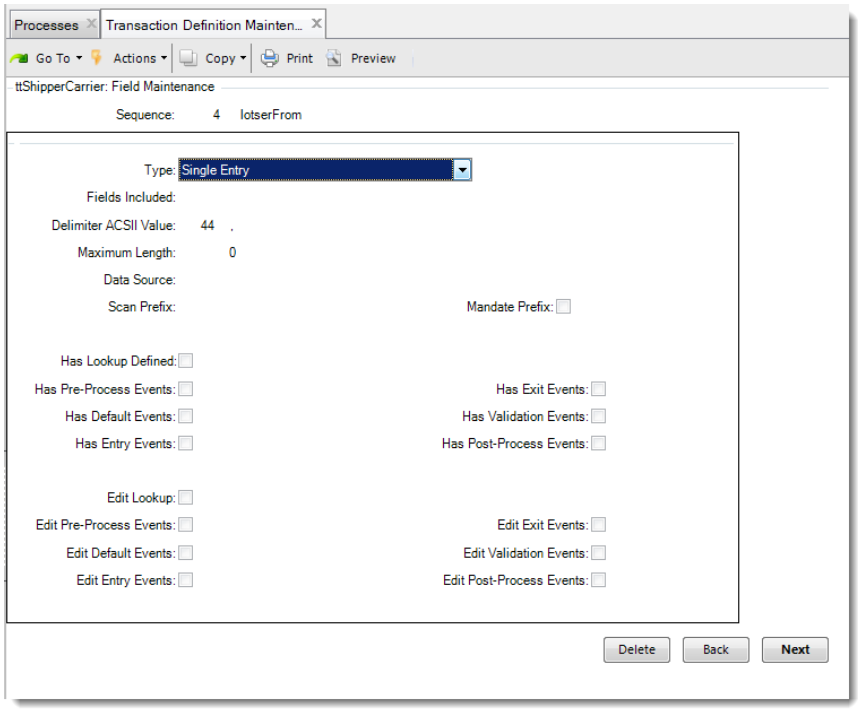
Fig. 3.7
Field Extents



Maintain Field Details

Press Next to enter another field frame to define specifics about the field, including the type, delimiter ASCII value, maximum length, source data, and any scan prefixes. You can specify whether lookups are defined for the field and whether you can edit the lookups. You can also indicate when there are associated pre- and post-events, validation, or default events, and specify when these can be edited.

Fig. 3.8
Field Details



Type. Specify the field data entry type:

1: Single entry (standard)

2: 2D

3: 3D

4: Packed - Fixed Length

5: Packed - Delimited

Fields Included. This field is display only and lists the fields included with this dataset buffer. You can modify this field only when the field data entry type is not 1: Single Entry. You can enter multiple field names here whose values are scanned together in this field.

Delimiter ACSII Value. Enter the delimiter ASCII character of the field delimiter. You can modify this field only when field data entry type is not 1: Single Entry. This is the separator for multiple field values when scanned together. By default this is 44 (ASCII code for comma).

Maximum Length. Enter the maximum length of the data.

Data Source. Indicate the QAD database field that is the source for this field's data.

This works in combination with the buffer query string. Based on the QAD table mentioned in the buffer query string, the buffer fields can have a data source to store the database table record value of the records retrieved from the query string.

Scan Prefix. Enter the scanning prefix for a field. This is useful, particularly for RF device users who prefer to scan field values from a label or pre-printed list, which typically have a prefix for identifying a value. For example, an item number can be printed as PART-1234. So the scan prefix is PART-. When users scan PART-1234, the system only considers the value following the prefix.

Has Lookup Defined. This field is informational and tells you when a lookup has been defined previously.

Has Default Events. This field is informational and indicates when the field has default events.

Has Validation Events. This field is informational and indicates when the field has validation events.

Has Pre-Process Events. This field is informational and indicates when the field has pre-process events.

Has Post-Process Events. This field is informational and indicates when the field has post-process events.

Edit Lookup. Indicate Yes to have users edit the lookup for the field. When Yes, the system displays a frame to let you enter the sequence of the lookup and the lookup name and specify whether the lookup is active or has details. “Create Lookups” on page 69

Edit Default Events. Indicate Yes to have users edit the default event. These are events that can occur when setting default values for a field. When Yes, the system displays a frame to let you enter the sequence of the default events and specify whether they are active or have details, similar to other detailed setup flow and screen.

Edit Entry Events. Indicate Yes to have users edit the entry events for the field.

Edit Exit Events. Indicate Yes to have users edit the exit events for the field.

Edit Pre-Process Events. Indicate Yes to have users edit pre-process events. These are events that are processed before the default event of the field. When Yes, the system displays a frame to let you enter the sequence of the pre-process event and specify whether it is active or has details; see “Create Events” on page 56.

Edit Validation Events. Indicate Yes to have users edit validation events. These are events that are processed after data is entered in the field. When Yes, the system displays a frame to let you enter the sequence of the validation event and specify whether it is active or has details, similar to other detailed setup flow and screen. When the validation event returns any value other than a logical Yes, the system creates a validation error. When a validation error occurs, users cannot proceed past the current field in which the error occurs, unless the user enters a valid value.

Edit Post-Process Events. Indicate Yes to have users edit post-process events. These are events that are processed after the system processes the field. When Yes, the system displays a frame to let you enter the sequence of the post-process e.vent and specify whether it is active or has details; see “Create Events” on page 56.

Edit Parameters

Once you choose the type of event and method, you move on to define the parameters for the type of event. In the same way that you entered a sequence number for the transaction, you can enter a sequence number for the parameter in the Parameter Maintenance Frame. For a method like a program function, the system defaults parameters based on their definition in Program Definition Maintenance (6.15).

Important The fields within the Parameters Maintenance frame vary, depending upon the event method you enter. For example, the parameters for a query-based differ from parameters for a program-based event; however, the initial setup is the same for all types, using these steps:

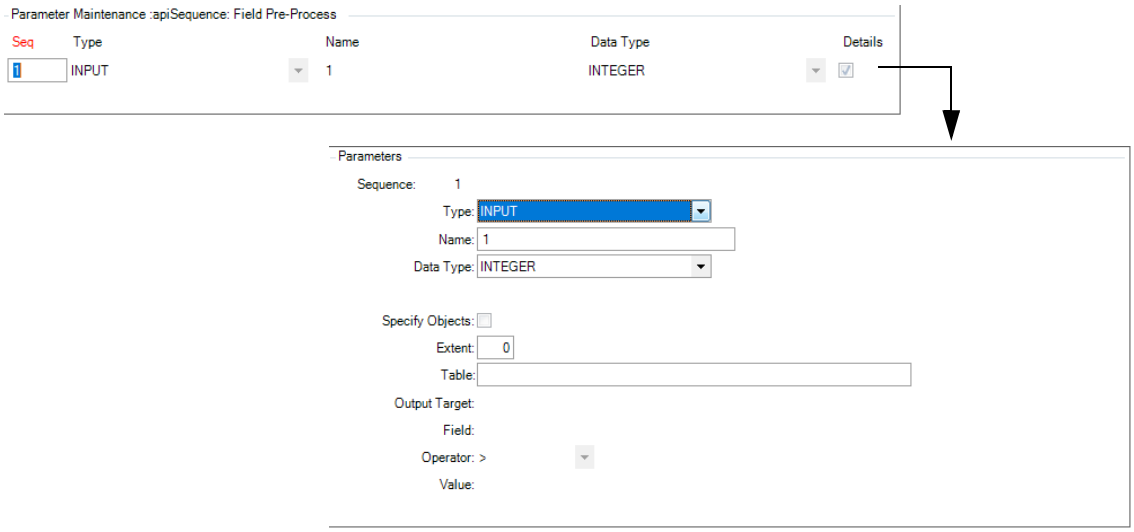
1 Specify the sequence of the parameter; press Enter.

2 Specify the Type, Parameter name, and Data Type.

3 Indicate Yes for Details to edit the record.

The system displays the Parameter Maintenance Frame.

Fig. 3.9
Transaction Definition Maintenance, Parameter Maintenance Frame



Sequence. Enter the sequence number for this event's parameters.

Type. Enter the type of parameter; choose from:

- BUFFER
- COMPARE
- CONSTANT
- FIRST
- INPUT
- INPUT-OUTPUT
- LAST
- OUTPUT
- RETURN
- Reposition
- SET

Note A buffer is a Progress parameter that the system cannot pass; however, it is presented because parameters that are brought in with the program require recognition by the system.

Name. Specify the name of the parameter. You enter the variable name to be applied.

Note the following:

- When you specify a field name value as part of the input, you enter the field name, and the system passes the value of the field when the field's value is available; otherwise, the system passes a field name as a string.
- When you want to pass the name of the field as the input parameter, put the field name in single quotes; for example, when you want to pass the field name for the Nbr field, enter `'Nbr'`; the system passes the field name, not the value of the Nbr field.
- When you want to pass a dataset as a parameter, you need to know the dataset configuration of the parameter:
  - CONFIG-DATASET: Transaction definition configuration dataset
  - RESPONSE: Response dataset
  - DATASET: API dataset
  - UI-CONFIG: UI dataset

Data Type. Enter a valid data type. Choose from:

- CHARACTER
- DATASET-HANDLE
- DATE
- DATETIME
- DATEIME-TZ
- DECIMAL
- HANDLE
- INT64
- INTEGER
- LOGICAL
- LONGCHAR
- MEMPTR
- N/A
- RAW
- RECID
- ROWID
- TABLE-HANDLE
- WIDGET-HANDLE

Details. Indicate Yes to view parameter details. The system displays an additional frame to enter details; see Figure 3.9 on page 54.

Sequence. The sequence number for this event's parameters.

Type. Displays the parameter type. The system defaults the parameter name as entered. You can edit and modify the parameter type selected. See "Type" on page 54.

Note The methods are described for the Method field to set up the event; see "Method" on page 59.

Name. The system defaults the parameter name as entered. You can edit and modify the parameter type selected. See "Name" on page 54.

Data Type. The system defaults the data type as entered. You can edit and modify the parameter type selected. See "Data Type" on page 55.

Specify Objects. From the object being returned by the method, specify which attributes/objects to retrieve values from and the target for its value.

When this field is selected and the data type is Object, another frame will appear where you can specify the values. The same object can be retrieved multiple times with different targets. For more information see, "Specify Objects" on page 63.

Extent. Enter the number of extents for the field. If the field has one or more extents, the system displays the field extent frame (see Figure 3.7). You should enter the sequence number for every extent to configure its functionality. The field extent has specific events that you can configure.

Table. Indicate a valid table name to run the event against; for example, if you specify a query as the method, enter the table name within the database or the dataset. For the Reposition method, the system defaults the table name as the current buffer name.

Output Target. Enter a valid field name in the dataset where the resulting value should be set.

Field. Enter a valid field name within QAD EE for data. For example, if you entered a query named PT_Domain, enter the field the system queries, such as **`pt_domain.`** When the system cannot find the field, it ignores the field you specify and instead considers it as a character value.

Operator. Enter a valid operator to be used with the Field and Value fields. For example, if you want a domain to equal the login domain, set Field to `pt_domain`, set this field to `=`, then set Value to **`LoginDomain.`**

Value. Enter a value for the field. You can hardcode a value, such as 0001, or enter a variable value, such as GlobalDomain, GlobalSite, and so on. When the system cannot find the field, it ignores the field you specify and instead considers it as a character value. Note, though, that global variables, such as a user ID, can change when the system runs the API, and it may be easier to use a variable like LoginDomain or GlobalDomain that does not change throughout the transaction.

Note For character values, when fields do not exist with the same name as the value, you can enter values without quotes. When fields exist with the same name, the value should be in quotes. Also, you can enter numeric values without quotes.

Create Events

You can create events for a transaction or you can modify events that are already part of an event template that you imported and use in a transaction.

You can create the following types of events:

- **Pre-process.** Specify events that run as part of pre-processing; for example to populate fields, initialize temp tables, or run any functions that must occur before the transaction runs. Pre-process events are the first type of event that the system runs when you enter the level (dataset, buffer or field) where they are defined.
- **Post-process.** Specify events that run after a successful commit. These are typically clean-up tasks, such as clearing out tables that were created temporarily during the commit or processing. Post-process events are the last type of events that the system runs when you enter the level (dataset, buffer or field) where they are defined.
- **Default.** Specify events that run as part of default data; for example, the system defaults the LoginSite in the Site field. These events populate a value in a field and are available only at field level.
- **Validation.** Specify events that run as part of data validation; for example, the entered `site` value is not a valid value in the QAD database. The system runs validation events after users enter or scan data; then, press Go. These events do not let users proceed further unless they enter a valid value and are available only at field level.
- **Commit.** Commit events run when you commit the transaction. When you run a commit event, you should run the commit event across the AppServer because this connection lets the system know when record locking occurs; that is, enter AppServer in the RunWhere field

when you set up parameters (see “Transaction Definition Maintenance, Parameter Maintenance Frame” on page 54.) for a commit event. These events are available at buffer and dataset levels only.

And you can create the different types of events for any of these levels:

- **Dataset level.** You can create pre-process, post-process, and commit events at the dataset level.
- **Buffer (temp table) level.** You can create pre-process, post-process, and commit events at the buffer level.
- **Field level.** You can create pre-process and post-process events, but not commit events at the field level; however, you can create other events, like default and validation events at the field level.
- **Lookups.** Lookups are at the field level, but you can create events for lookups, too. For example, you can create a pre-process event to build a workfile to generate the query against the ABS master and explode the ABS master to find items on the shipper. Then, in the post-process event, you can delete the workfile records you created.

Event-Creation Flow

The flow of creating or modifying the events is the same, regardless of the event type or level. The system presents fields that let you know that events are already a part of the transaction, such as Has Pre-Process Events, Has Post-Process Events. The fields are informational only, as you can create new events or edit the existing events for the transaction by specifying Yes to the Edit event fields.

For each level of the transaction that you edit—dataset (top level), buffer, field, or lookup—when you indicate that you want to edit events, and indicate the type of event you want to edit (pre-process and so on), the system displays a series of screens that let you:

1. Enter a sequence and description.
2. Indicate if the event is active or has details.
3. Go on to enter detailed information.

Important The screens that display to create/edit the events are similar, regardless of the type and level of event you are creating/editing. For this reason, the topics that follow walk you through editing one type of event at a particular level.

Specifying the Method for the Event

Once you click the field to edit an event—regardless of the level you are editing within the transaction—then finish all fields in the frame and press Go, the system presents a screen to let you edit the event.

The event editing begins with a frame that displays the sequence and description fields and two fields to indicate whether the event is active and whether you want to edit the event. Events that are already attached to a dataset, field, and so on have the sequence set, but you can change the sequence in this frame.

Use the following procedure to:

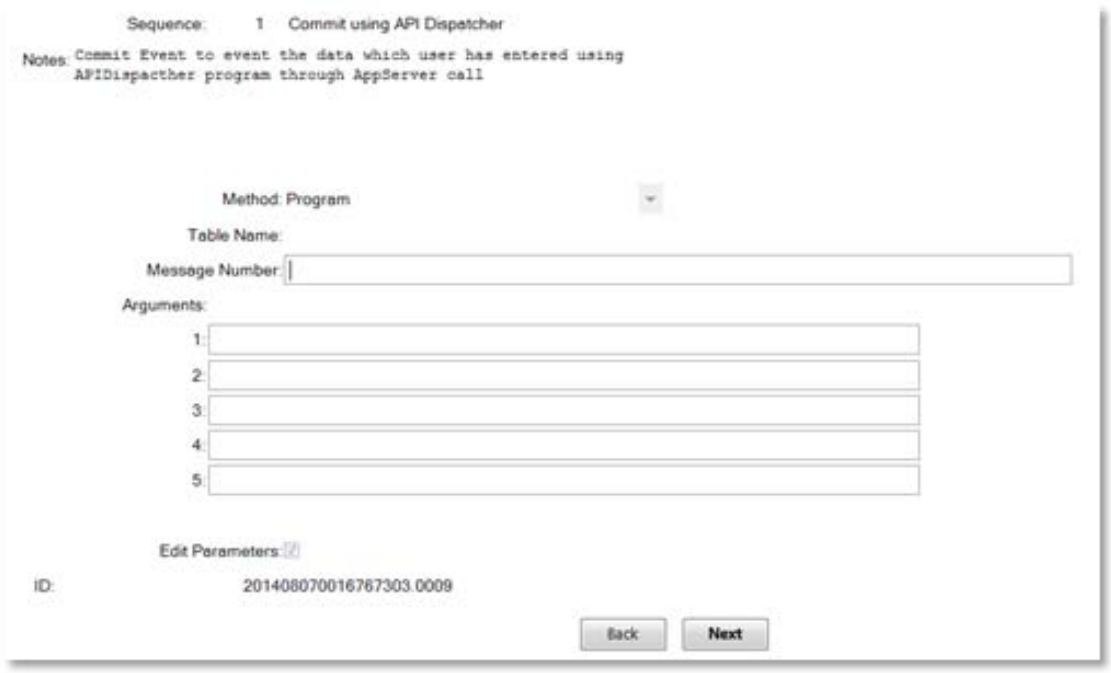
1. Specify the sequence number; press Enter to specify the description.
2. Indicate if the event is active and whether you want to edit details; press Go.

   The system displays a frame to enter details about the event.
3. Optionally, enter any notes about the event.
4. Specify the method.
5. Specify the table name and message number (using the table name depends upon the method of the event; for example, for a query method event, you specify a valid QAD EE table name that the system queries when the event runs.)
6. Indicate whether the parameters are editable. When Yes, the system displays a Parameter Maintenance frame with the transaction and dataset name.
7. Specify the message number. You can enter up to five arguments for the message number specified.
8. Indicate Yes in the Edit Parameters field to edit parameters.

   When Yes, the system displays a frame to enter the parameter name, datatype, and details.

The following figure depicts a Commit Event set up for a Query type method.

Fig. 3.10
Create Events, Method Specification



Sequence. Enter a sequence number for determining the order of processing of this event during the transaction.

Code. Enter a unique alphanumeric code for the event. The code can be assigned to one or more events and you can enable or disable them at the same time based on criteria.

Description. Enter a description of the pre-process dataset.

Active. Indicate if the event is active. When No, the event never occurs in the transaction.

Details. Enter to edit details.

Notes. Optionally, enter any notes for the pre- or post-process dataset.

Method. Enter the method for the event.

- **Can-Find Query.** This method is set much the same way as the Query method but with some differences. Enter this method to check for the first occurrence of a record based on the input parameters you specify. It does not return field values from the record or raise error conditions when the record does not exist. It returns a value of Yes (when a record is found) or No (when a record is not found) to a field of your choosing in the dataset. Valid types of parameters for this method are Input and Set.
- **Constant.** Enter an identifier whose associated value cannot typically be altered during transaction execution. You typically use constants for fields. For example, set a default event at the field level with a Constant type with a Parameter Name as a specific PO number, such as `PO-0001` for a pre-process event so that the field is defaulted with the PO number.
- **Function.** Enter a program's function to perform some action, such as retrieving a value.

  Note that if the Event Method is Function, the user cannot add or delete parameters, change the Type, or change the Data Type.
- **Get Record Count.** This event method returns the total number of records returned from a specified query.
- **Initialize Fields.** For future enhancements. Not available yet.
- **Pre-Fetch Data.** Specify pre-fetch data for this event method when you want the system to request that the event data before the data is actually needed for the buffer.
- **Program.** Enter the name of a program to run, such as a program that obtains the total weight of items on a record. Once you enter the program, you can choose which procedure to run.

  Note that if the Event Method is Program, the user cannot add or delete parameters, change the Type, or change the Data Type.
- **Query.** Enter a query to perform validations, set values within a record or for a field, and so on.
- **Reposition.** Enter Reposition to move to an existing record within the buffer. The system uses the buffer key values to identify unique buffer records and move to the first or last record in that buffer. Users can traverse through the records using Page-Up and Page-Down keys.
- **Script.** Enter a script to run.
- **Update Record Count.** This event method updates up to three fields in a UI dataset table. The values returned include:
  - The total number of records generated from the query
  - The actual record number

- A concatenation of the two numbers: <*Record Number* > - < *Total Number of Records*>

Table Name. Indicate a valid table name to run the event against; for example, if you specify a query as the method, enter the table name within the database or the dataset. For the Reposition method, the system defaults the table name as the current buffer name.

Message Number. Specify a message number for a message to display when this event happens. This is in addition to any messages for the programs that were called. You can specify up to five arguments for the specified message number. The argument values replace the number sign (#) in the message text.

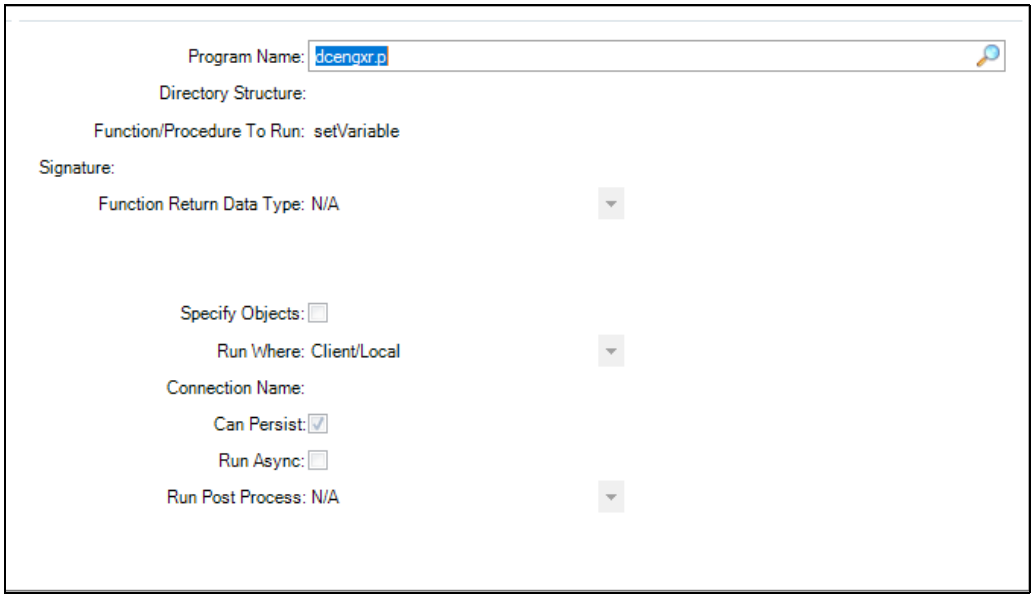
Edit Parameters. Specify Yes to edit parameters. The system presents a frame to enter the sequence number and description of the parameter, and indicate if the parameter is active or has details. The system then displays the Parameter Maintenance frame; see “Transaction Definition Maintenance, Parameter Maintenance Frame” on page 54.

Parameters for a Program-Method Event

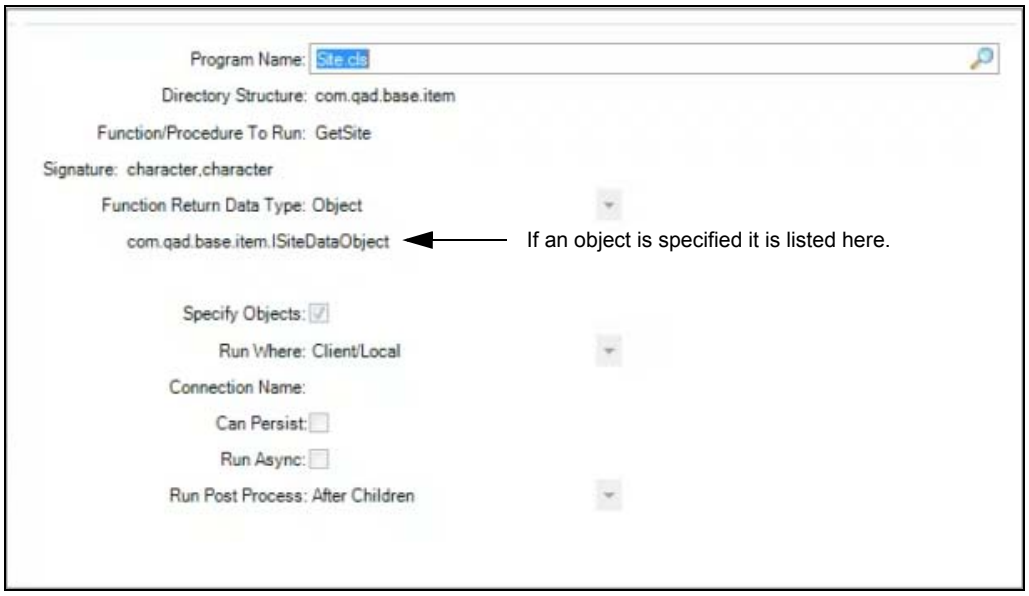
When you enter a Program as the Method for an event, the system displays a frame for you enter program-specific data for the event.

Fig. 3.11
Program Specifics for a Program-Method Event

Program



Class File



Program Name. Enter a valid QAD EE program name (.p) whose definition is imported in the system.

Directory Structure. Enter the path to the file on the disk that you want to run. When this field is blank, it is assumed that the directory structure is `us/<xx>`, where <*xx*> represents the first two characters of the file. For example, the directory structure for `sosomt.p` is: `us/so`. In this example, this field would be left blank.

An example where this field would be populated is for `APIDispatcher.p` because the directory structure for this program is `com/qad/erp/api.` In this example, you would enter `com/qad/erp/api` in this field.

Function/Procedure to Run. Enter a function within the program to use. For example, you can enter a GetVariableValue function if it exists in the program to retrieve the value from anywhere in the transaction from any buffer to get the value.

Signature. Identifies the signature for the method.

Note This field is part of the index for the program details because class files can have the same method name defined multiple times within the same file with different signatures. It consists of the parameter sequence number, a parameter type identifier, and the data type identifier for each parameter in the signature. When being displayed to the user in a readable format, each parameter data type is displayed, separated by a comma. When the parameter is a type of an output, then OUTPUT or INPUT-OUTPUT is displayed before to the data type.

This field is display only.

Function Return Data Type. Enter the return data type from the function. See the description for “Data Type” on page 55.

Specify Objects. From the object being returned by the method, specify which attributes/objects to retrieve values from and the target for its value.

When this field is selected and the data type is Object, another frame will appear where you can specify the values. The same object can be retrieved multiple times with different targets. For more information see, “Specify Objects” on page 63.

Run Where. Indicate where you run the transaction. Choose from:

- **AppServer.** Application server connection.
- **COM.** Communication interface, such as RS232.
- **Client/Local.** The client, or local machine, when connecting to a host. This requires no setup.
- **FTP.** File Transfer Protocol address to transfer files.
- **HTTP.** Hypertext Transfer Protocol for the Web.
- **SCRIPT.** The name of a script used to connect; for example, an AcquireConnection that casts the connection.
- **TCP.** Transmission Control Protocol, a protocol within the Internet protocol suite.
- **WebService.** A web hosting service, and Internet hosting service that makes a website accessible via the World Wide Web.

Connection Name. Enter the name of a valid AppServer service that exists in Service Connection Maintenance (6.22.1).

Can Persist. Enter Yes if the program can persist. The value of this field defaults from settings in Program Definition Maintenance; see “Define the Program” on page 22.

Run Async. Specify Yes to run the program asynchronously.

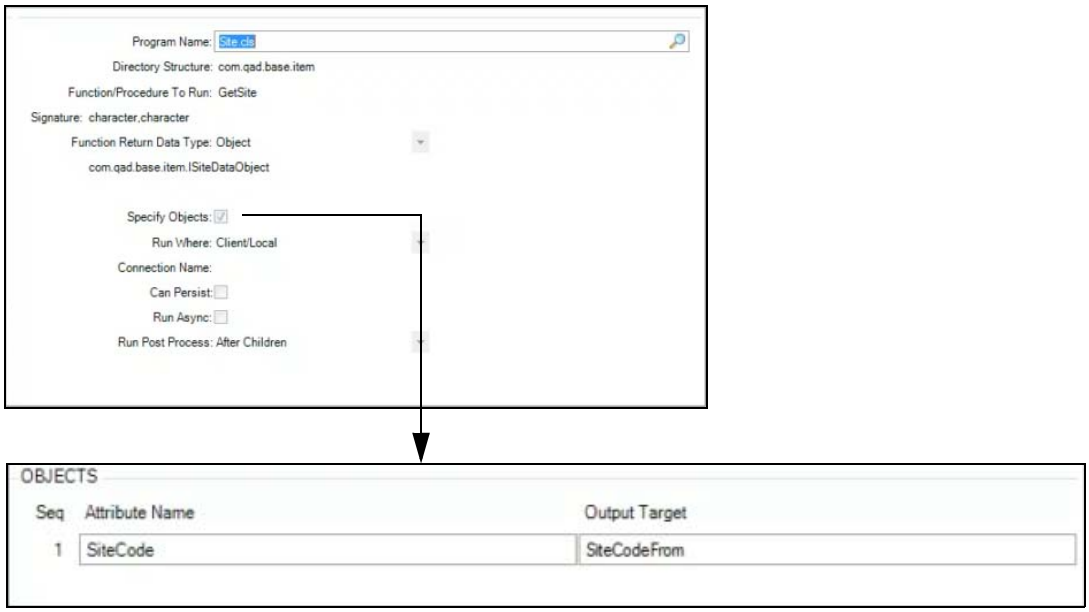
Run Post Process. Define how and when this event runs before, after, or both before and after the children run. This field is not applicable for Commit events.

Message Number. Enter a number for a new message number to display when the event has a failure. This message is in addition to messages provided with existing programs. You can specify up to five arguments for the specified message number. The argument values replace the number sign (#) in the message text.

Specify Objects

When the Specify Objects check box is selected and the data type is Object, the Objects frame appears where the user can specify the values. The same object can be retrieved multiple times with different targets.

Fig. 3.12
Specify Objects Frame



Seq. Enter a numerical value to identify the record. This value must be greater than zero.

Attribute Name. Enter the name of the attribute to retrieve from the object. This field cannot be blank and is not validated against the object.

Output Target. Enter the name of a field in the transaction dataset to place the attributes value. This value is not validated against the transaction definition.

Parameters for Other Method Types

The flow of screens varies, depending upon the Method you enter. Previous topics discussed a Query method and the Program method. The following table describes other Method types and the flow of screens, based on the type.

Table 3.1
Parameter Field Explanations for Event Methods

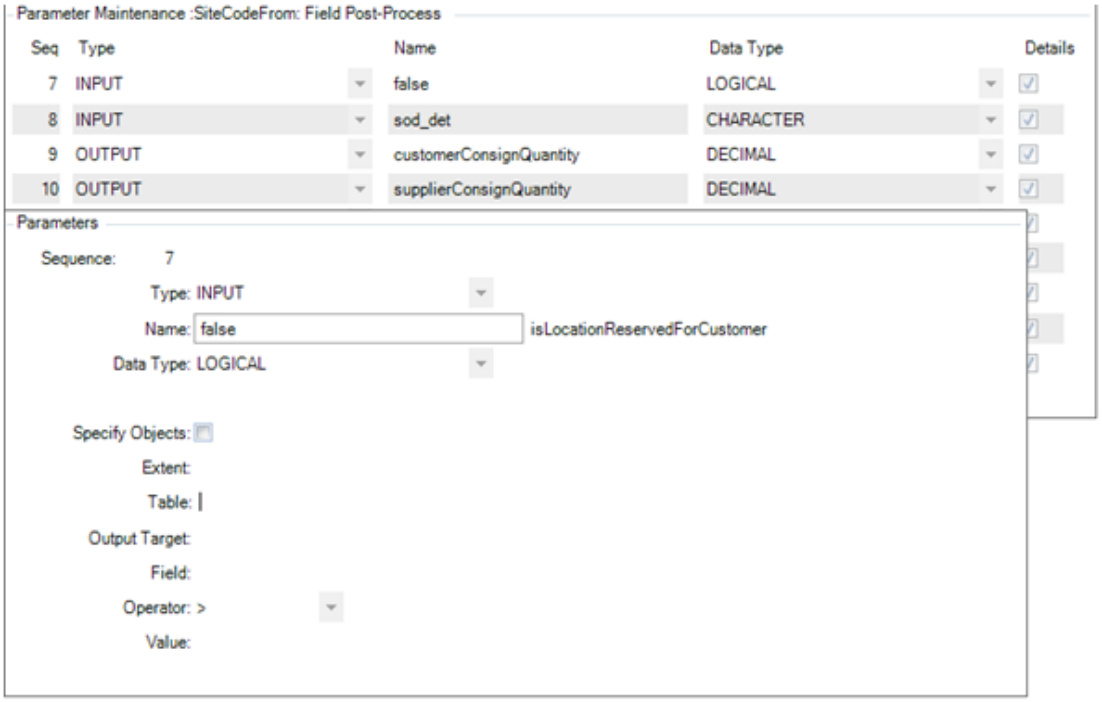
Event Method	Description
Can Find Query	Enter a valid parameter type, such as Input or Set; see “Method” on page 59.
Constant	Enter Constant as the Method, go on to Edit Parameters; see “Transaction Definition Maintenance, Parameter Maintenance Frame” on page 54.
Function	Enter function as the method, the system displays the same screen as for Program Method; see Figure 3.11 on page 61. Note that for functions, the variable name parameter and datatype parameter are needed. When you specify a specific function or procedure for a program, the system brings in all input, output, and other previously defined parameters for the function/procedure.
Get Record Count	Enter Get Record Count as the method, go on to Edit Parameters; see “Transaction Definition Maintenance, Parameter Maintenance Frame” on page 54.
Pre-Fetch Data	Enter Pre-Fetch data as the method. The system displays the same parameter screen as for Program Method; Figure 3.11 on page 61
Query	See the description in “Specifying the Method for the Event” on page 57.
Reposition	Enter Reposition as the Method, go on to Edit Parameters; see “Transaction Definition Maintenance, Parameter Maintenance Frame” on page 54
Script	Enter Script as the Method, go on to Edit Parameters; see “Transaction Definition Maintenance, Parameter Maintenance Frame” on page 54
Update Record Count	Enter Update Record Count, go on to Edit Parameters; see “Transaction Definition Maintenance, Parameter Maintenance Frame” on page 54.

Event Parameter Details

You can edit parameter details for the events when you set the Details field to Yes.

You are first prompted to set up the sequence, parameter name, and data type for each parameter; then, specify whether parameter details display. When you set the parameter Details field to Yes, the system displays the Parameter Details frame; see “Edit Parameters” on page 54 for field descriptions.

Fig. 3.13
Event Parameter Details



Example Event Creation

The following topics present two examples that show you how to set fields for:

- An event that sets a default value for the Nbr field based on a counter

  For the example, there is a counter on the transaction and you want the default number to be one plus the counter. So in the example, the number in the field is incremented by one each time the transaction runs.
- An event that gets the last PO number in the system for the Nbr field

Example Default Nbr field based on a counter:

Event Level: Field level event

Event Type: Default event

Transaction: Nbr

Set the following fields, which display in different frames.

Field Details Frame (see Figure 3.8):

Edit Default Events: Yes

Field Maintenance Frame (see Figure 3.6):

(Field) Sequence: 1

Description: Default Value for Nbr using a constant

Active: Yes

Details: Yes

Method: Query

Table Name: po_mstr

Type: Constant

Parameter Name: PO-002345 (the default value)

Data type: Character

Parameter Details Frame (see Figure 3.13):

Extent: 0

Field: Nbr

Operator: =

Value: 1 + ctr

As another example, you can set up a query method, and then go on to set parameters so that the query gets the last PO number, sets other criteria, and indicates which field to return the value to as shown in the following parameter setup:

Fig. 3.14
Query Parameter Setup Example

Seq	Type		Parameter Name	Data Type		Details
1	LAST	[V]	po_domain	CHARACTER	[V]	no
2	INPUT	[V]	po_site	CHARACTER	[V]	yes
3	INPUT	[V]	po_status	CHARACTER	[V]	yes
4	RETURN	[V]	po_nbr	CHARACTER	[V]	yes

As you can see, you can continue to build by adding different types (INPUT, OUTPUT, and so on), and then using the details frame to further customize the event.

Create Event Templates

There are two ways to create a template in Data Collection:

- Save your event creation data to a template by exporting the data in the .NET UI version of Data Collection.
- Use Event Template Maintenance (6.22.3) to set up a template for programs, functions, and so on, that the system calls as an event process.

Export Data in .NET UI

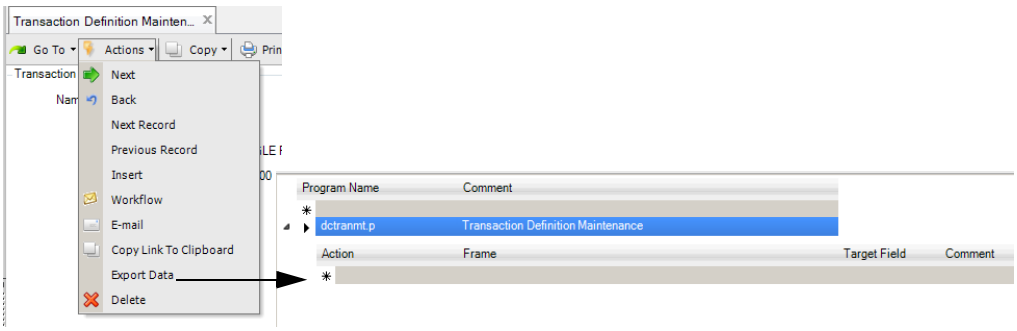
When you create events in the .NET UI version of Transaction Definition Maintenance, you can save the event data that you create to a template. When you create the event data, from the .NET UI menu:

1. Click Action in the .NET UI menu.
2. Select Export Data

   The system displays an Export frame with the program name defaulted.
3. Enter the Action, Frames, and Target fields to export.
4. Press Go.

You can view or edit the template using Event Template Maintenance (6.22.3).

Fig. 3.15
Export Data



Event Template Maintenance

You can create an event template for:

- Events for a buffer (table), lookup, or field
- Pre-process, post-process, or commit events

You can set up the method for the event, then drill down to create or edit parameters for the method you choose. Use the following procedure as a guideline for editing the event template.

1 Enter the template name.

The system displays the sequence number and description.

2 Press Enter to edit the description and indicate if the template is Active or if you want to view/edit details.

When you do not select Details, the system displays the Sequence field and description for you to modify or delete.

3 Press Enter to add notes associated with the template.

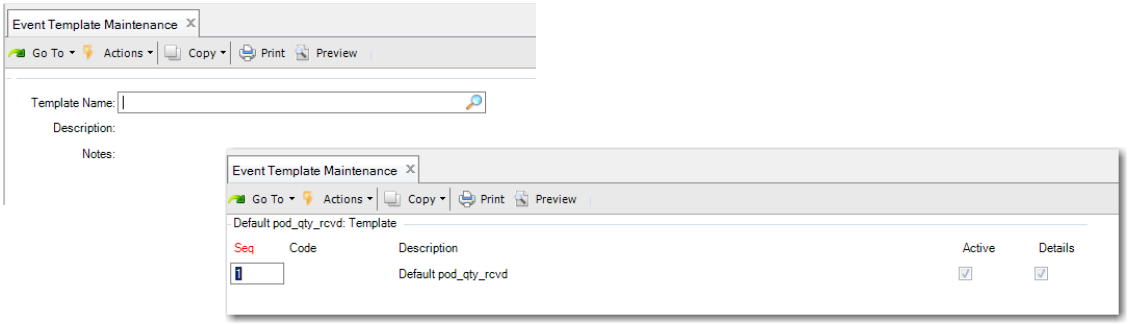
4 Enter the method (program, script, query constant, field lookup, and so on.)

5 Enter other method details, such as the name, function/procedure to run, function return data type, where to run the function, the connection name and more: see “Parameters for Other Method Types”.

6 Enter the message number and select whether to edit parameters; see “Edit Parameters” on page 54.

A series of frames display that vary depending on the type of event—pre-process, and so on; whether the event is for a buffer, field, and so on; the method, and whether you choose to edit parameters. The frames that display are similar to those for setting up the event in Transaction Definition Maintenance. Refer to “Create Events” on page 56.

Fig. 3.16.
Event Template Maintenance



Template Name. Enter the template name.

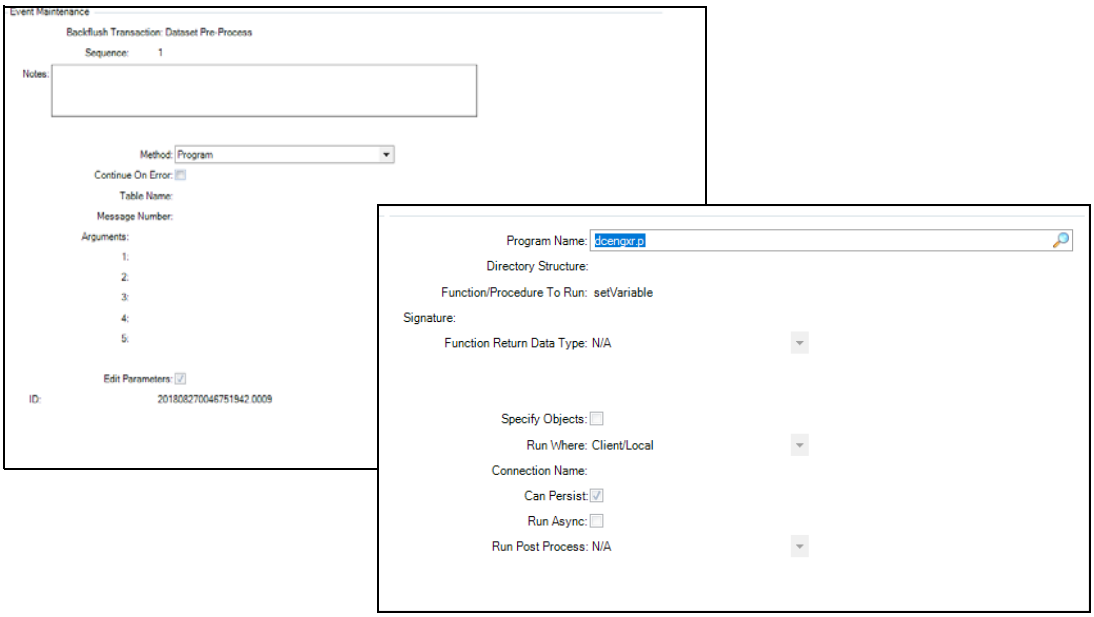
Description/Notes. Optionally enter a description and any notes associated with the event template.

Active. Indicate Yes if the template is active.

Details. Indicate Yes to display template details. When Yes, the system displays additional screens for you to modify template data.

When you select to view or edit details, the system displays an additional frame, as shown in Figure 3.17.

Fig. 3.17
Event Template Maintenance, Method Selection and Edit



The fields in this screen are similar to fields within Transaction Definition Maintenance to set up the event method; refer to “Specifying the Method for the Event” on page 57.

Create Lookups

You create lookups in Transaction Definition Maintenance at the field maintenance level. Lookups can have events attached to them. You can also edit the lookups that:

- You imported into the program through lookup templates; see “Import/Export Programs” on page 26.
- Came with other dataset/program data you use in the transaction

In Transaction Definition Maintenance, you first encounter fields to edit lookups when you set up the field data for a dataset buffer; see Figure 3.8 on page 52.

When you set the Edit Lookups field in the screen shown in Figure 3.8, then finish the field detail screen, the system displays a frame to set up the lookups; see Figure 3.18.

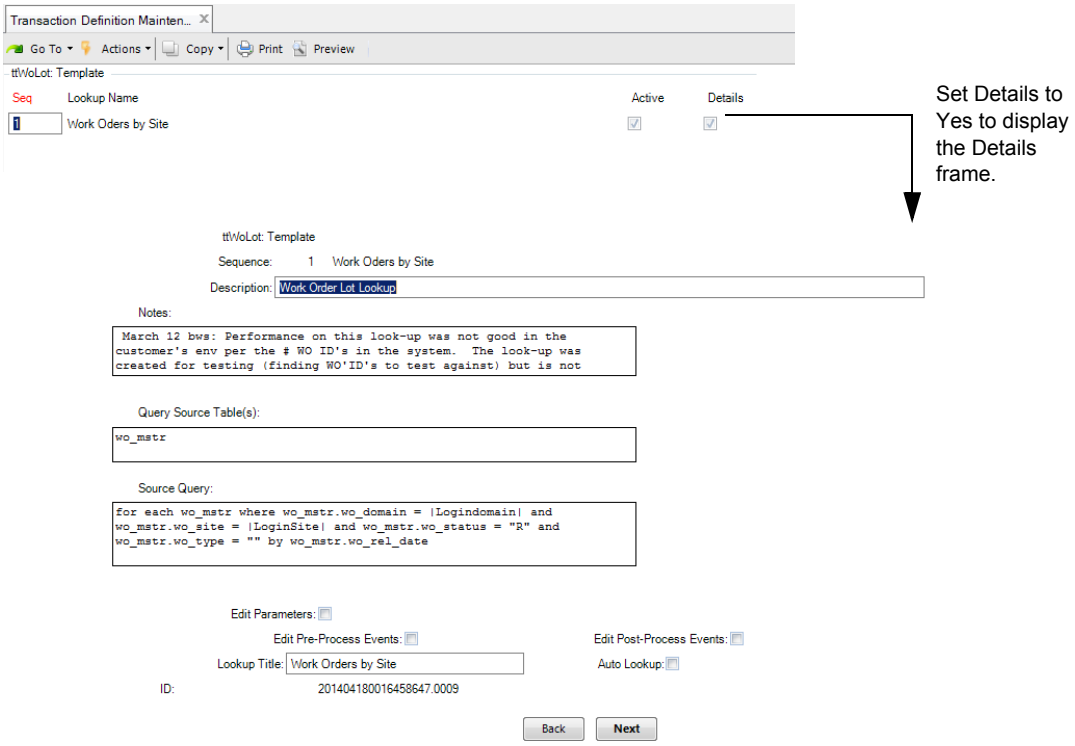
When you set up lookups, the flow is the same as that for other elements that you set up in Transaction Definition Maintenance, except that you can drill down into additional lookups once you edit details of a single lookup. For example, if you set up a lookup for a PO number and you set up the query details of that lookup, you can next create a lookup to see the items on that PO once the user selects the PO number, so the sequence in the creation process lets you drill down into the lookup for the PO editing parameters. Then, you can edit parameters and drill down into the lookup for the items.

Use the following as a general guideline to create lookups:

1. Start by entering a sequence and description.
2. Indicate if the item is active or has details.
3. Go on to enter detailed information about the source tables and source queries.
4. When you are passing parameters as a field name, edit field lookup and field columns data.
5. Indicate if events are attached, and edit event data through the event editing screens.
6. Edit parameters.

You can save your created or modified lookups to a template, too; see “Create Lookup Templates” on page 73.

Fig. 3.18
Transaction Definition Maintenance, Lookup Maintenance



Seq. Enter the sequence number for the lookup.

Code. Enter a unique alphanumeric code for the lookup event. The code can be assigned to one or more lookups and you can disable and enable them at the same time based on criteria.

Lookup Name. The system defaults the lookup name. Press Enter to change the name.

Active. Indicate if the lookup is active.

Details. Indicate if you want to view/modify details. When Yes, a frame displays the Query Source Table and the Source query (code); see “Event Parameter Details” on page 64.

Description. Enter a description of the lookup definition.

Notes . Optionally, enter any notes associated with the lookup.

Query Source Table. List the source tables to query in the order you want to follow in the query. Separate a list with commas. For example, enter po_mstr, po_dets.

Source Query. Write the query in this field. The source query must be prefixed with the table name. Also, when passing in parameters, the source must be in between pipe symbols (|), as shown in the following example of data to enter for the Source Query:

```
for each po_mstr where po_mstr.po_domain = |LoginDomain| and
po_mstr.po_site = |LoginSite| and po_mstr.po_status <> “”
```

Lookup Title. Enter the title of the lookup; for example, `PO Master.`

Edit Parameters. Indicate Yes to edit parameters. When Yes, an additional frame displays that lets you specify if the parameters in the lookup query should refer to a dataset buffer field or its parent lookups. The system does not prompt for global values.

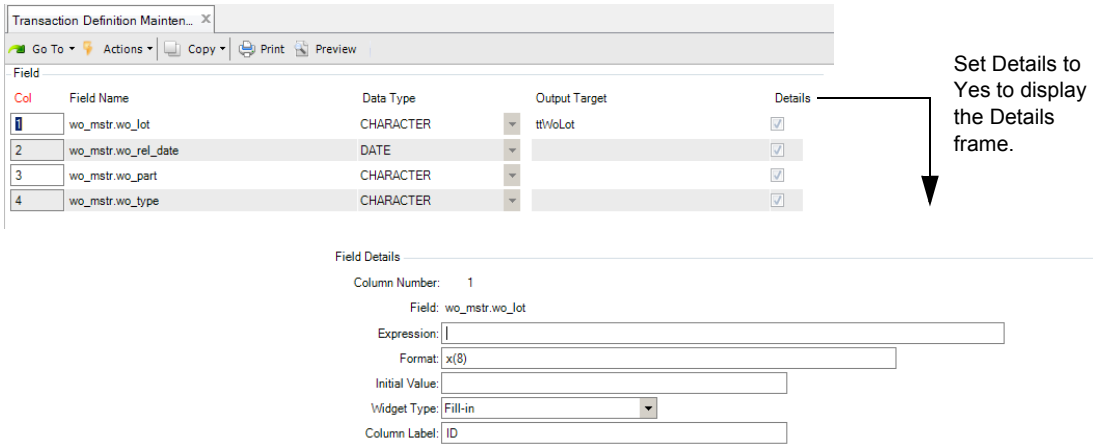
Edit Pre-/Post-Process. When you select Yes to edit pre- or post-process events, the flow and frames are similar to those that display when you select to edit pre-/post-process events in Transaction Definition Maintenance; see “Create Events” on page 56.

Auto Lookup. Indicate Yes to automatically display the lookup prior to entering the field for the lookup. The lookup displays first unless the user already entered a value for the field. When No, the user presses an additional key—such as F2 in character QAD EE—to display the lookup.

Define Columns for the Lookup

To create lookups, you go on to define the columns for the lookup, including specifying the data type, output target, and other details.

Fig. 3.19
Transaction Definition Maintenance, Lookup Maintenance



Col. Enter the sequence for the columns/

Field Name. Enter the field name for the columns. When the field name is a database field, it must be prefixed with the table name, as in `po_mstr.po_nbr`.

When you enter an expression, you must surround the expression with the pipe symbol (|), for example, `|ABC|`.

Data Type. Set the data type. Choose from:

CHARACTER	INT64
DECIMAL	DATETIME
LOGICAL	INTEGER
DATE	

Output Target. Specify the field to which the system returns the looked-up data.

Details. Specify Yes to edit details.

Expression. Enter a logical expression and the source for data that the system attaches to the value that displays in the column or for doing a logical calculation for a character datatype field; for example, when you want to add a prefix or suffix to the data in the column. You must use the pipe symbol (|) in the expression, as shown in the following example that adds the PO status to the PO number:

|po_mstr.po_nbr| + |po_mstr.po_status|

Format. Enter the format of the field, or widget, and so on. So, for example, the format for PO Number would be `x(0)`.

Initial Value. Enter an initial value for the column.

Widget Type. The only supported widget type is Fill-in.

Column Label. Enter a label for the column to be presented to the user, such as `PO Nbr.`

Create Drill-Down for the Lookup

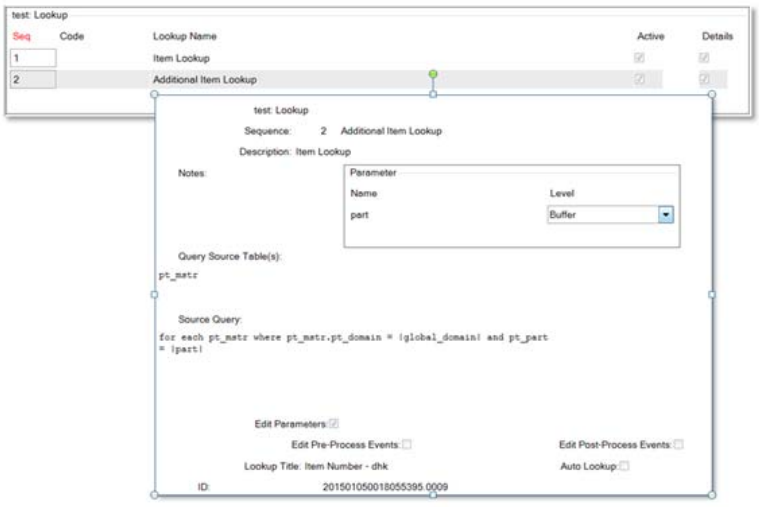
You can create a drill-down lookup (multi-level lookup) by configuring two or more active lookups. When users press the Help key on the field in a mobile or RF device, the system displays the active lookup first. Pressing Help again displays the next lookup. The drill-down lookups continue displaying until there are active lookups.

Drill-down lookups can have lookup queries based on the dataset fields or their parent lookup (except for the first sequential lookup). To create the drill-down for the lookup, you:

- Should have parameters in the lookup query, for example, `part` which is a buffer field name or you should have a parent lookup column name
- Set Edit Parameter to Yes.

  When you set Edit Parameters to Yes, the system displays an additional frame that lets you specify when the parameters in the lookup query refer to a dataset buffer field or its parent lookups; see Figure 3.20. The system does not prompt for global variables.

Fig. 3.20
Transaction Definition Maintenance, Drill Down for Lookups



Create Lookup Templates

Use Lookup Template Maintenance (6.22.4) to create a template for lookups that you can use in other transactions that you build. You can set up a template by modifying parameters, pre-process, and post-process data, and more.

Navigation

Note that the general flow of lookup template setup for most of the program follows the same flow as other Data Collection maintenance programs. That is, you drill down and enter pre-/post-processing data, parameters, and fields for the lookup, following this typical flow:

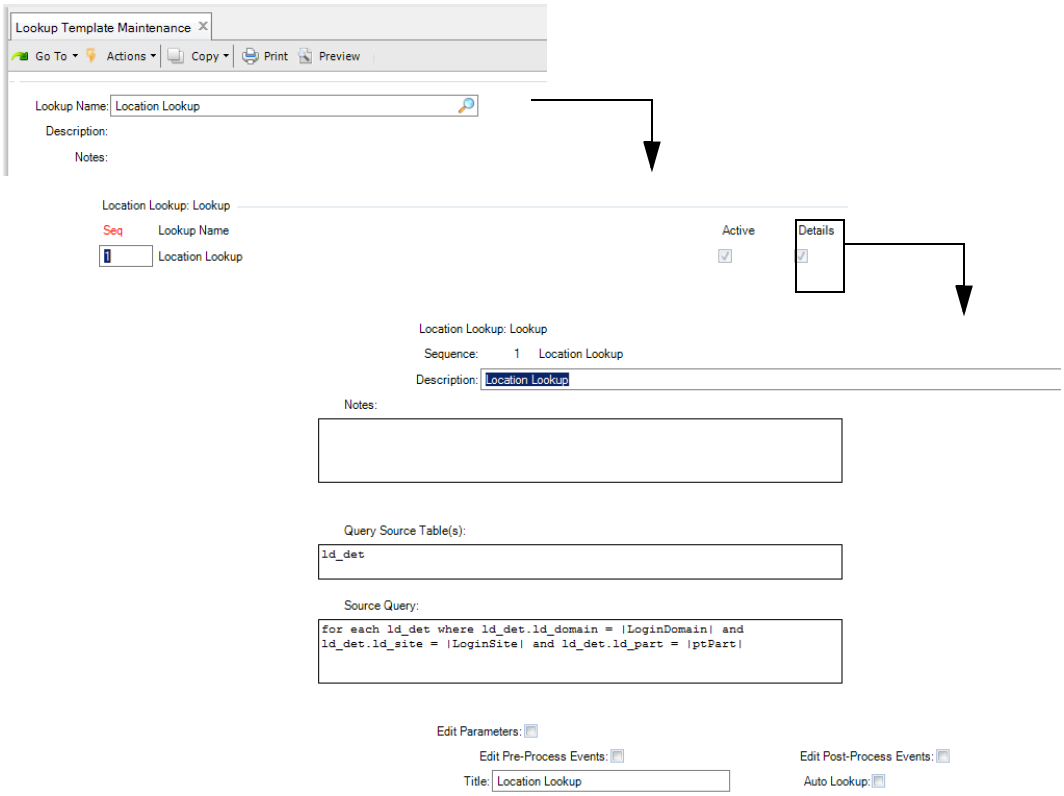
1 Start by entering a sequence and description.

2 Indicate if the item is active or has details.

3 Go on to enter detailed information.

Editing the lookup template follows the same flow and frames that are presented when you create the lookup in Transaction Definition Maintenance; that is, you create the sequence and define fields, enter details, then go on to create the sequence and define columns, and enter column details. The fields are similar to those when you create the lookup; see “Create Lookups” on page 69.

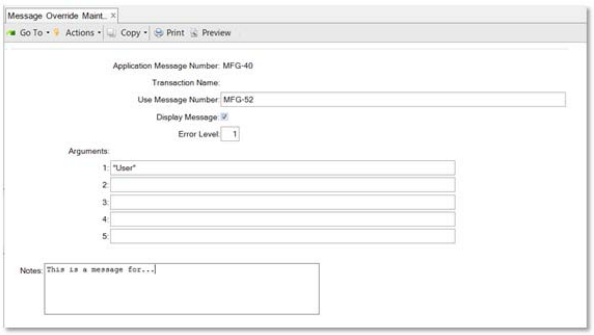
Fig. 3.21
Lookup Template Maintenance (6.22.4)



Create Message Overrides

Use Message Override Maintenance (6.22.13) to specify how the system processes an application message number when it receives the message during a Data Collection transaction. You can choose to ignore the message, change the message that displays, or change the error level that the message generates. You can optionally provide notes about the override.

Fig. 3.22
Message Override Maintenance (6.22.13)



Application Message Number. Specify the message number that the application returns. This field cannot be blank. Use the drop-down lookup to select a message number. The lookup displays message numbers from the dcMessage table.

Transaction Name. Specify a transaction definition name for which the message override occurs. When you leave this field blank, the system overrides the application message number for all transactions that use the override rules defined here.

Use Message Number. Specify the message number for the replacement of the application message number.

Display Message. Indicate Yes to display the message and No to ignore the message. When No, the system ignores the application message number when it is received and does not generate an error.

Error Level. Specify the error level for the message. This lets you override the error level that the application returns. When zero, the error level returned is an error.

Arguments. You can specify up to five arguments for the new overriding message number. The arguments are values to be used instead of the number sign (#) in the message text.

Publishing Transactions

When you publish a transaction, the system generates a transaction key for the transaction and any child transactions that the parent references. The system only generates transaction keys for those transactions that do not have keys associated with them. Transaction keys are required if you want to export a transaction.

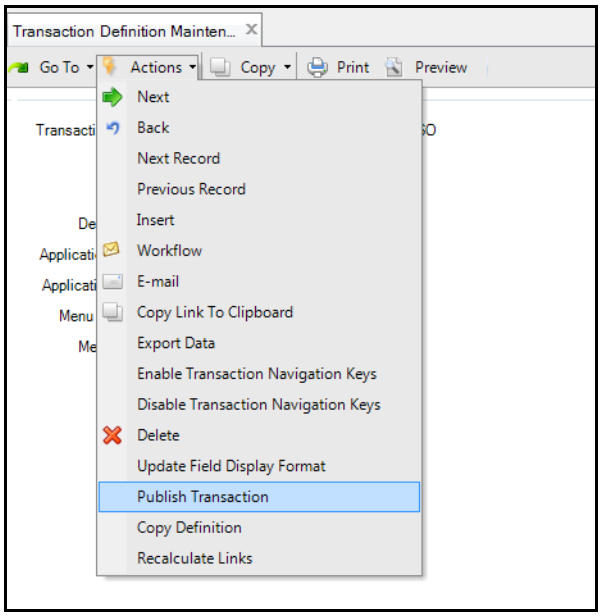
Note The following video, *Data Collection - Transaction Definition Publishing*, is now available. It covers the transaction definition publishing process, which is new functionality available in QAD Automation Solutions: Data Collection v3.1.

https://qad.mediasite.com/mediasite/Play/e64d64fa07a940c88981753882048d211d

To publish a transaction, following these steps:

1 Select Publish Transaction from the Action menu.

Fig. 3.23
Publishing a Transaction



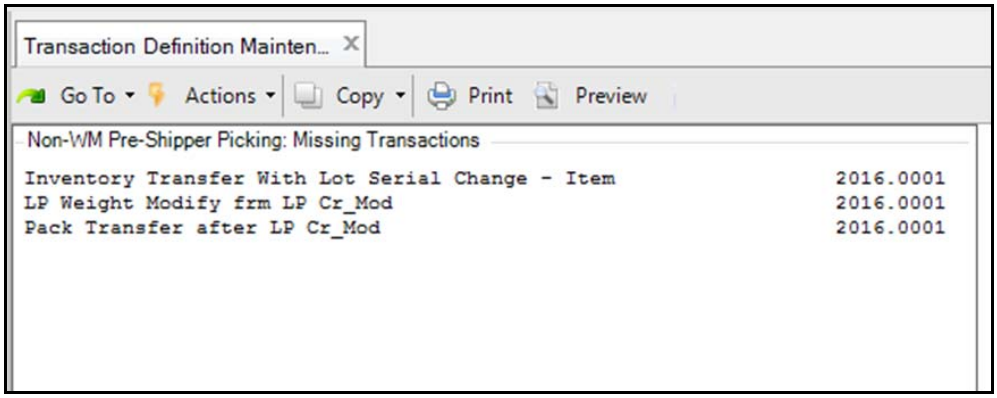
2 The system prompts you to confirm. Click Yes to continue.

Fig. 3.24
Publishing a Transaction

3 The system checks if there are any missing and unpublished child transactions:

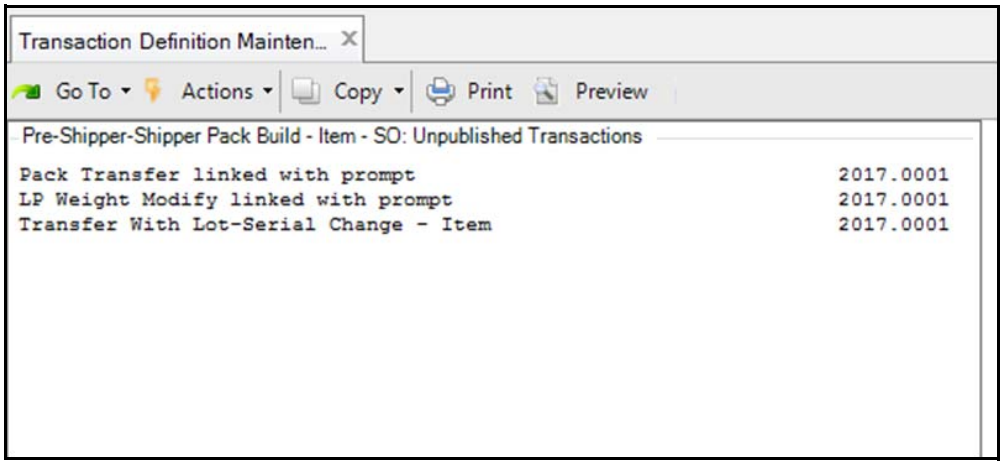
a The system checks if the parent transaction references any missing child transactions, and if so, it displays a list of those transactions. You cannot publish a parent transaction that has missing child transactions.

Fig. 3.25
Publishing a Transaction



b The system also checks if any of the child transactions have not been published, and if so, it displays a list of those transactions. When a parent transaction is published, all child transactions are published as well.

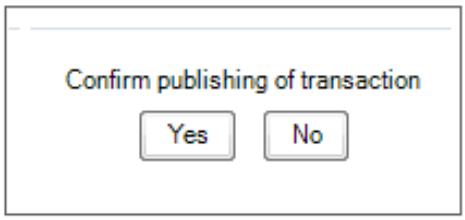
Fig. 3.26
Publishing a Transaction



4 After the system checks for missing and unpublished child transactions, the system prompts you to confirm. Click Yes to publish.

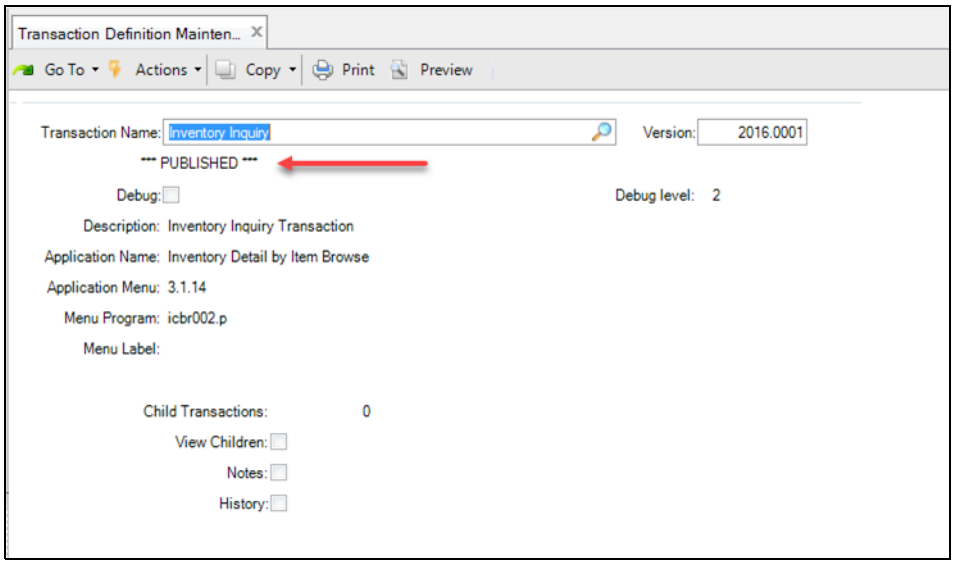
Note When the user selects Yes to publish the transaction, all child transactions that are not published are published along with the parent transaction. Each transaction will be assigned its own key.

Fig. 3.27
Publishing a Transaction



5 When a transaction is published, the transaction definition record indicates that it has been published.

Fig. 3.28
Publishing a Transaction



Editing a Published Transaction

To edit a published transaction, create a copy of the transaction and make the changes to the copied version. You can only copy a published transaction that have transaction keys. For more information about copying transactions, see “Copy Transaction Definitions” on page 84.

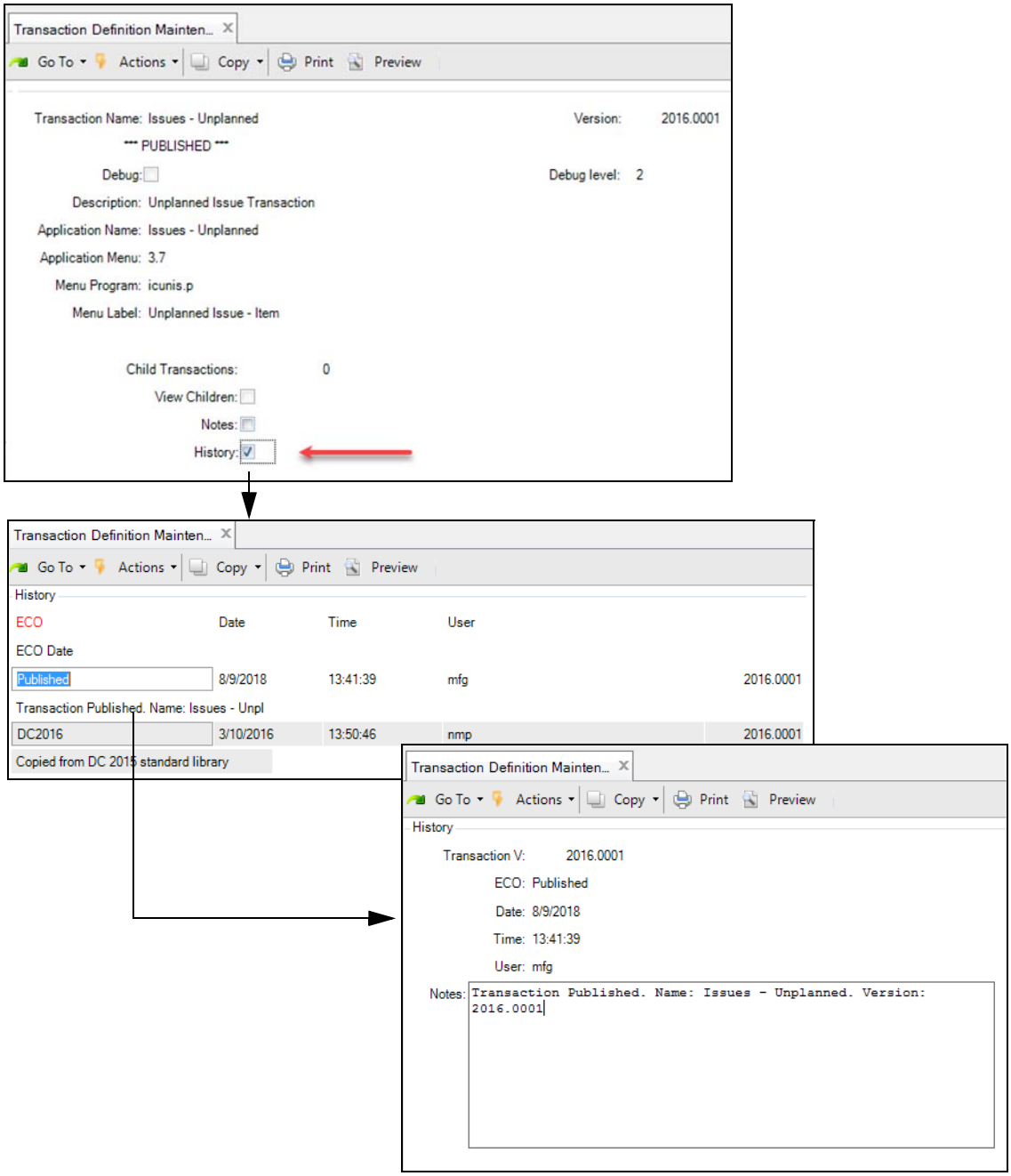
Note When you copy a transaction, a history record is created that states where it was copied to and where it was copied from. If you change the name of the transaction to something completely different, this allows you to trace back to see the original transaction it was copied from.

Note When you copy a transaction, all the history records that were associated with the original transaction are copied over to the new version.

Viewing the Publishing History of a Transaction

When a transaction is published, the system generates a history record. To view the historical publishing records, select the History check box in Transaction Definition Maintenance (6.16).

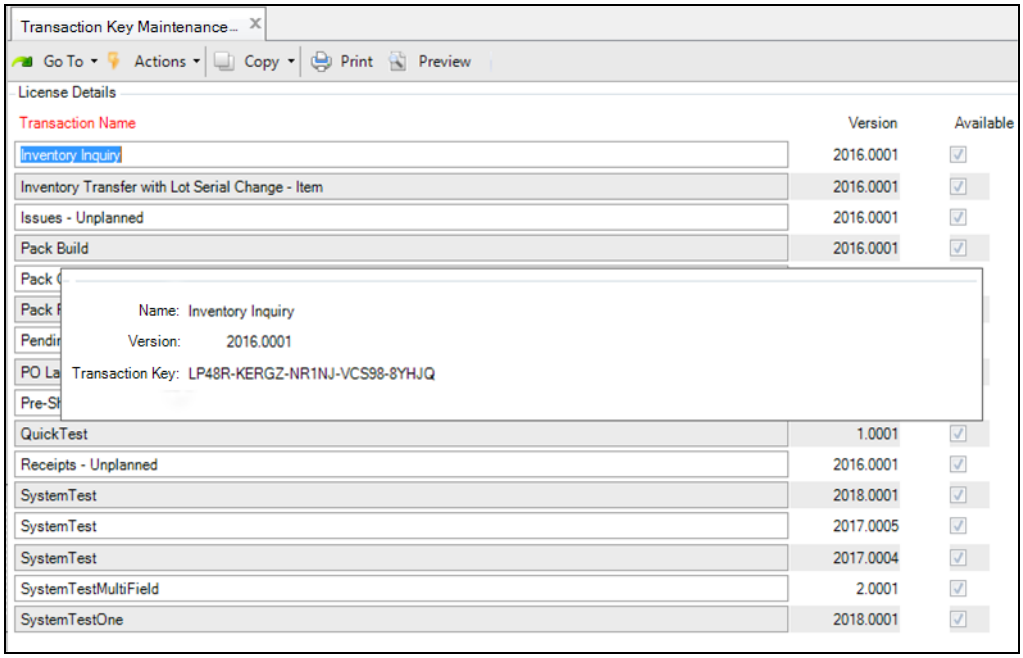
Fig. 3.29
Viewing the Publish History of a Transaction



Transaction Key Maintenance

Transaction Key Maintenance (6.22.17) allows you to view the transaction keys that have been created for the transactions. Only transactions that have been published have transaction keys. To view a transaction key, select the record.

Fig. 3.30
Transaction Key Maintenance



Name. Displays the name of the transaction definition.

Version. Displays the version of the transaction definition.

Transaction Key. Displays the transaction key assigned to the transaction definition.

If this field displays the transaction key, the transaction has been published.

If this field is blank and is not editable, the transaction has not been published.

If this field is blank but editable, the transaction has been published but a transaction key has not been entered.

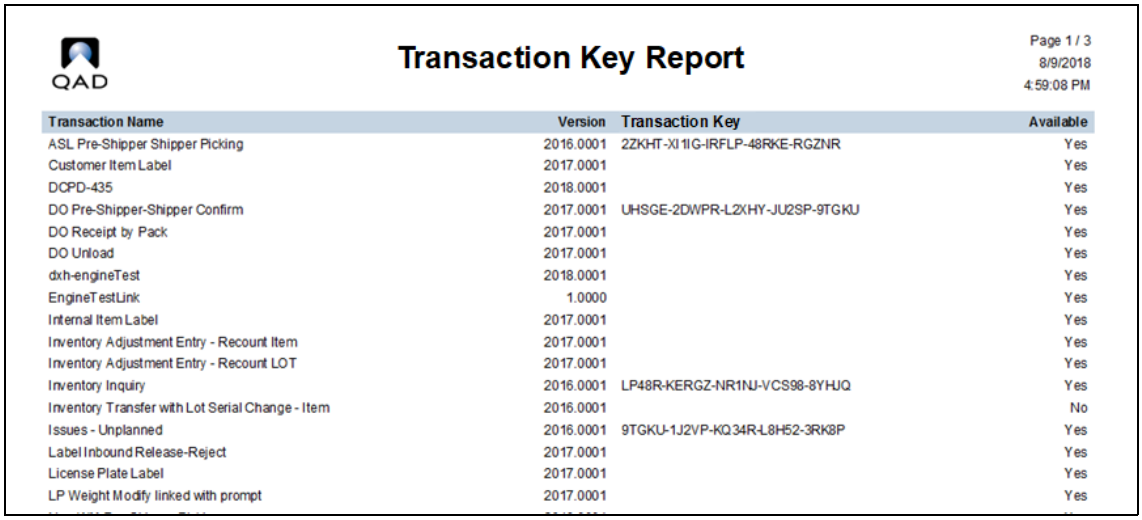
When a transaction is published, the system generates a transaction key for the transaction and any child transactions that the parent references. The system only generates transaction keys for those transactions that do not have keys associated with it. Transaction keys are required if you want to export a transaction.

Transaction Key Report

The Transaction Key Report (6.20.13) shows the transaction keys for each transaction. This report also shows every transaction name and version that has been in the system.

Note The Available field shows whether the transaction has been deleted or not. If Available = Yes, the transaction is currently in the system. If Available = No, the transaction was previously in the system but has been deleted.

Fig. 3.31
Transaction Key Report



QAD

Transaction Key Report

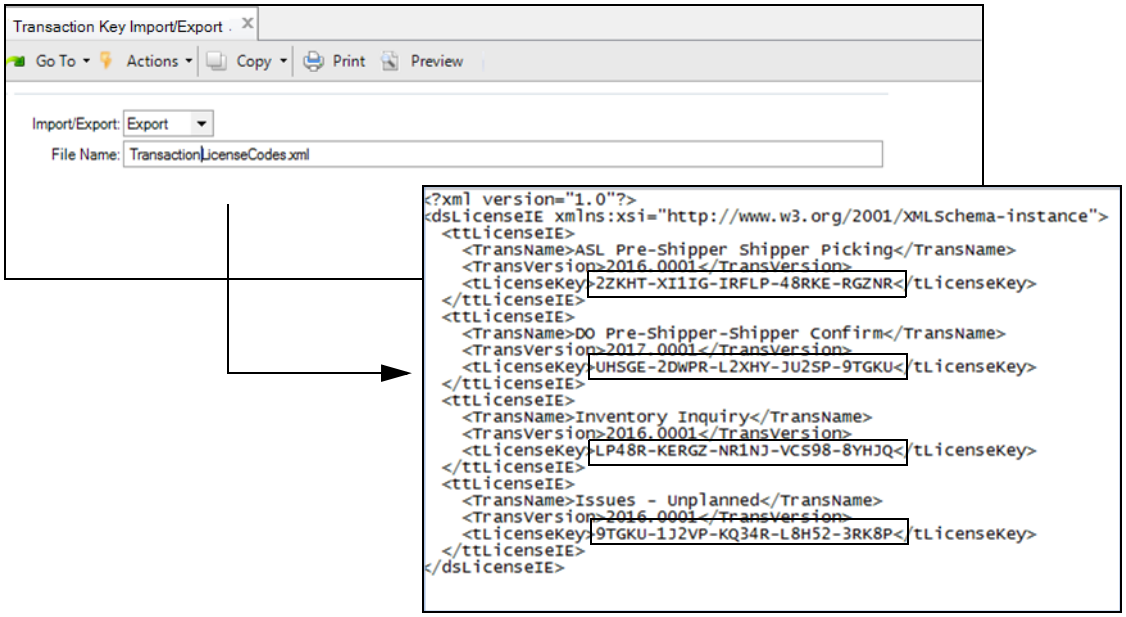
Page 1 / 3
8/9/2018
4:59:08 PM

Transaction Name	Version	Transaction Key	Available
ASL Pre-Shipper Shipper Picking	2016.0001	2ZKHT-XI1IG-IRFLP-48RKE-RGZNR	Yes
Customer Item Label	2017.0001		Yes
DCPD-435	2018.0001		Yes
DO Pre-Shipper-Shipper Confirm	2017.0001	UHSGE-2DWPR-L2XHY-JU2SP-9TGKU	Yes
DO Receipt by Pack	2017.0001		Yes
DO Unload	2017.0001		Yes
dxh-engineTest	2018.0001		Yes
EngineTestLink	1.0000		Yes
Internal Item Label	2017.0001		Yes
Inventory Adjustment Entry - Recount Item	2017.0001		Yes
Inventory Adjustment Entry - Recount LOT	2017.0001		Yes
Inventory Inquiry	2016.0001	LP48R-KERGZ-NR1NJ-VCS98-8YHJQ	Yes
Inventory Transfer with Lot Serial Change - Item	2016.0001		No
Issues - Unplanned	2016.0001	9TGKU-1J2VP-KQ34R-L8H52-3RK8P	Yes
Label Inbound Release-Reject	2017.0001		Yes
License Plate Label	2017.0001		Yes
LP Weight Modify linked with prompt	2017.0001		Yes

Importing/Exporting Transaction Keys

Transaction Key Import/Export (6.23.23) allows you to import or export a list of transaction keys that you have in your system as `.xml` files.

Fig. 3.32
Transaction Key Import/Export



Import/Export. Indicate whether you are importing or exporting files.

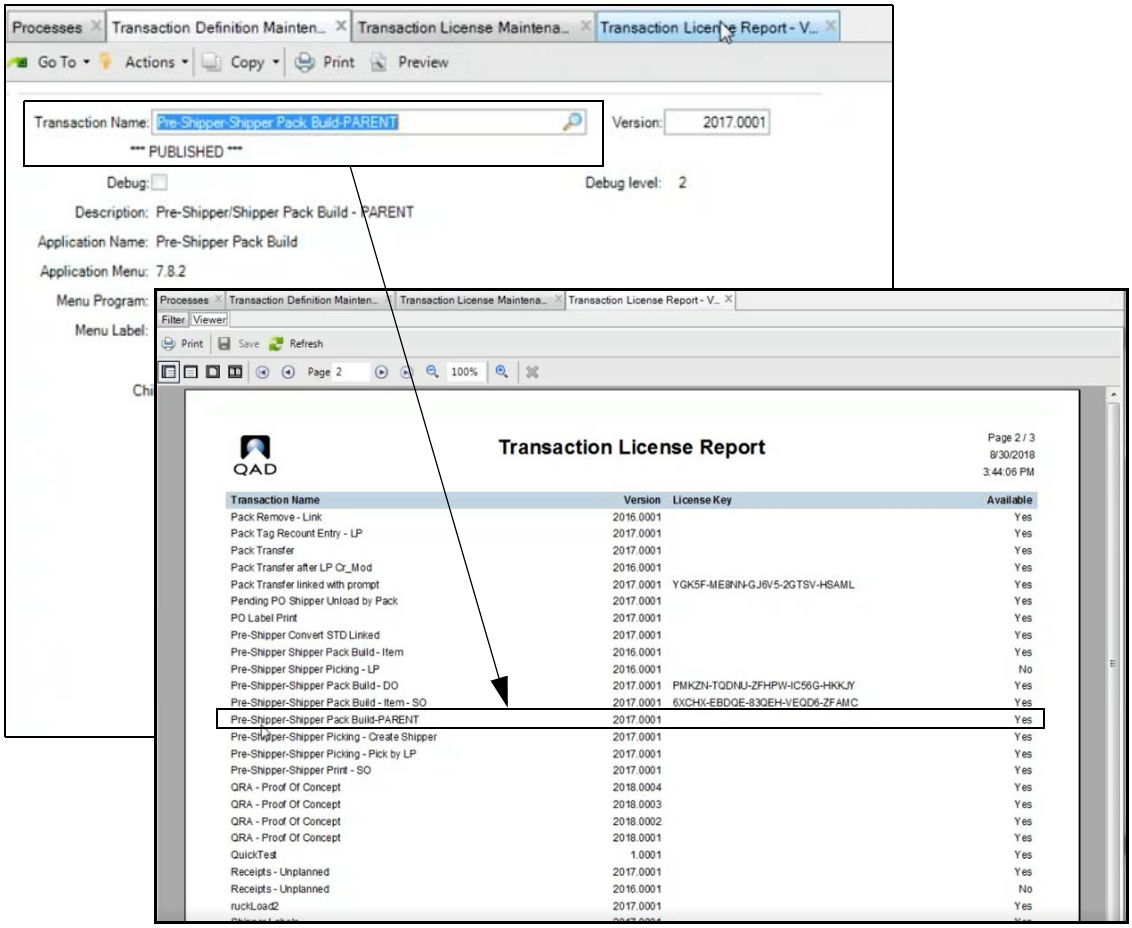
File Name. Enter the `.xml` file name of the file you are importing or exporting.

Manually Entering a Transaction Key

If you send a published transaction to another user, the user will need to manually enter the transaction key in Transaction Key Maintenance (6.22.17) to use that transaction.

For example, the following figure show a published transaction that does not have a transaction key entered.

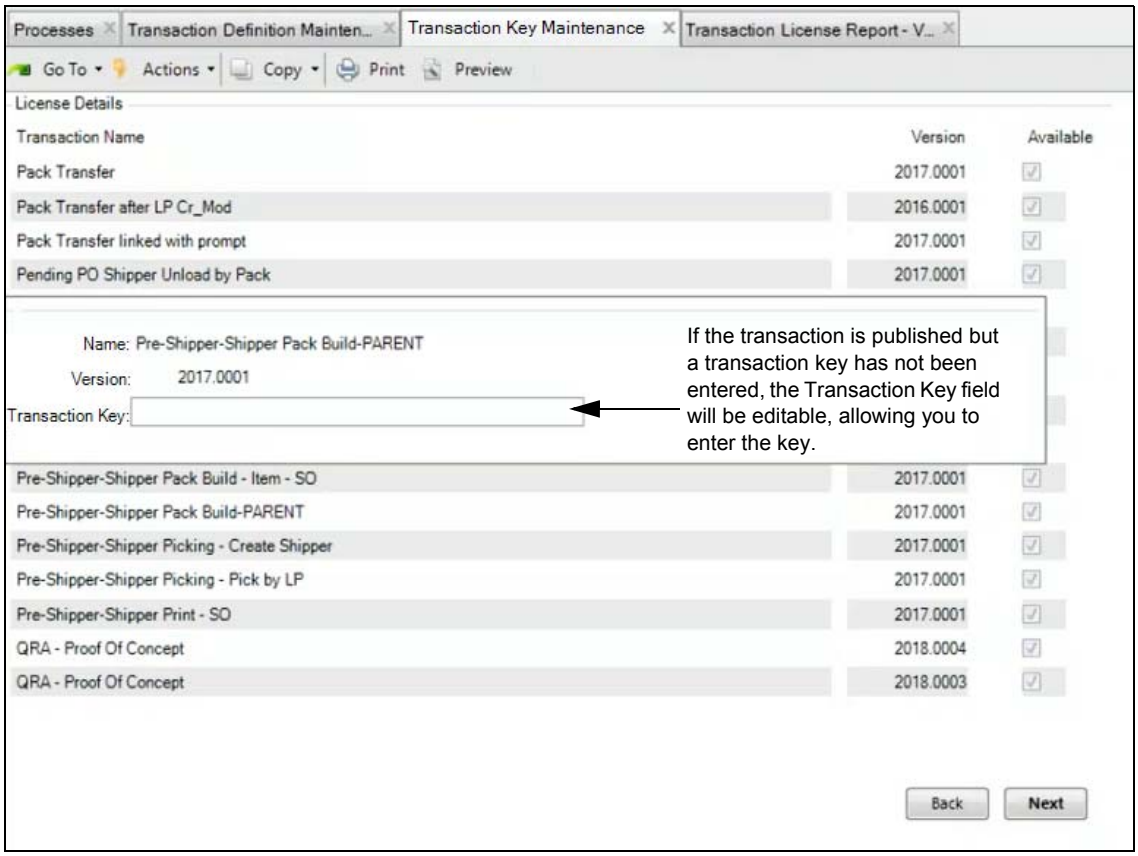
Fig. 3.33
Entering a Transaction Key



To manually enter a transaction key, open Transaction Key Maintenance (6.22.17), select the transaction, enter the transaction key in the Transaction Key field, and then click Next.

Note If the transaction is published but a transaction key has not been entered, the Transaction Key field will be editable, allowing you to enter the key. If the transaction is not published, the Transaction Key field will be blank and will not be editable.

Fig. 3.34
Entering a Transaction Key

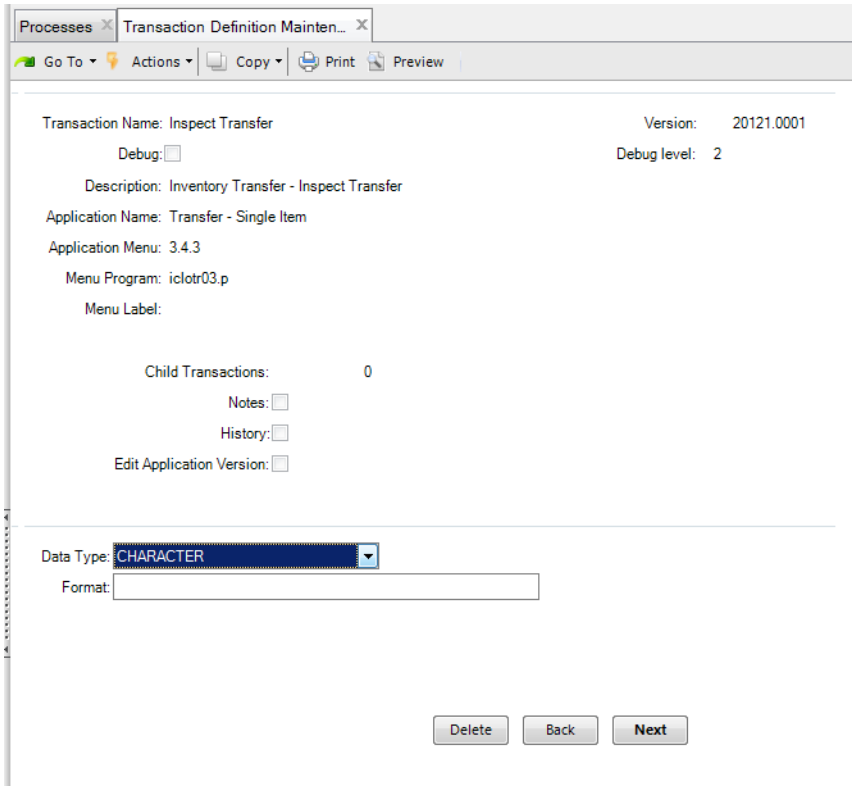


Transaction Definition Action Menu

Update Field Display Format

The Update Field Display Format action, located in Transaction Definition Maintenance (6.16), allows the user to reset the display format of fields for a transaction based upon the datatype. When this action is selected, the user specifies the data type and the new display format of the data, which is then applied to all fields for that data type within the given transaction.

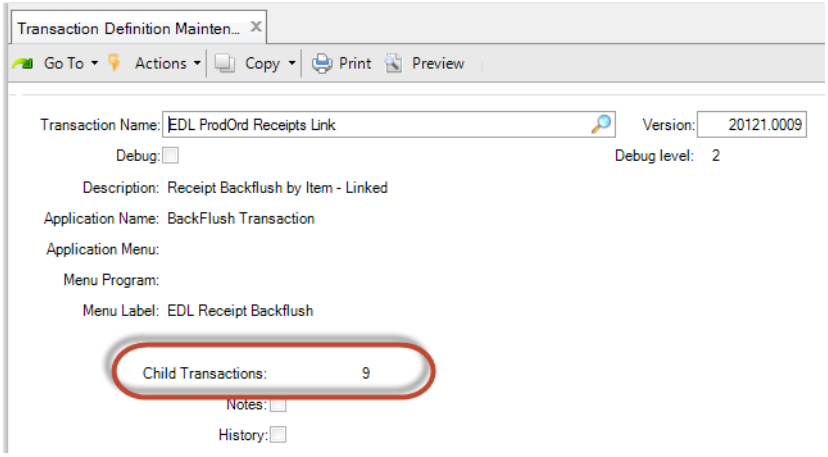
Fig. 3.35
Update Field Display Format



Recalculate Number of Child Transactions

The Recalculate Links action, located in Transaction Definition Maintenance (6.16), allows the user to manually recalculate the number of child/linked transactions that are called/executed from within the selected transaction. Typically this value does not need to be manually updated from the Actions menu because the maintenance routine keeps track and updates when a liked transaction is added or removed. The Child Transactions field displays the number of child/linked transactions that are called/executed from within the selected transaction.

Fig. 3.36
Recalculate Links



Copy Transaction Definitions

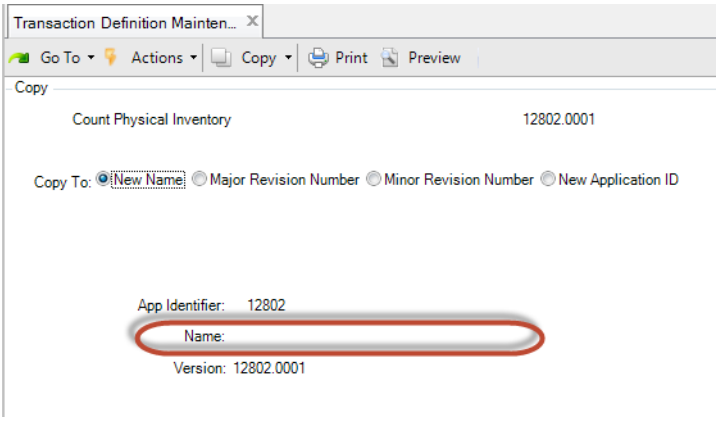
The Copy Definition action, located in Transaction Definition Maintenance (6.16), allows the user to create a transaction by copying information from a source transaction. When the Copy Definition action is selected, the user is given the following options on which information is copied to the new transaction:

- **New Name.** When this option is selected, the user enters a new name for the transaction.

  When the new transaction is created:

  - All notes are copied to the new transaction.
  - The history records from the source transaction are not copied over to the new transaction.
  - The application identifier is the same as the source transaction.

Fig. 3.37
Copy Definition - New Name

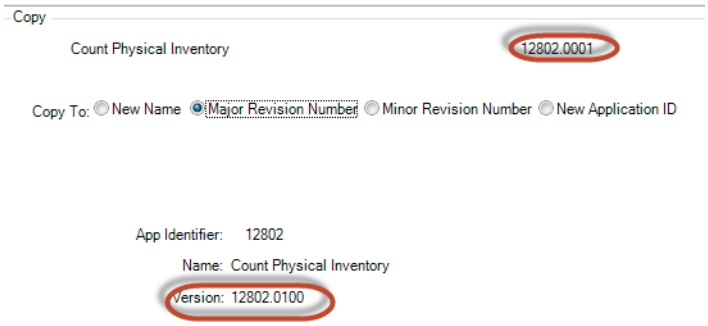


- **Major Revision Number.** When this option is selected, the major revision portion of the version number (the first two digits to the right of the decimal) is incremented by one. The minor revision portion of the version number (the last two digits to the right of the decimal) is set to zero.

  When the new transaction is created:

  - The transaction name for the new transaction is the same as the source transaction.
  - All notes and history records are copied to the new transaction.
  - The application identifier is the same as the source transaction.

Fig. 3.38
Copy Definition - Major Revision Number

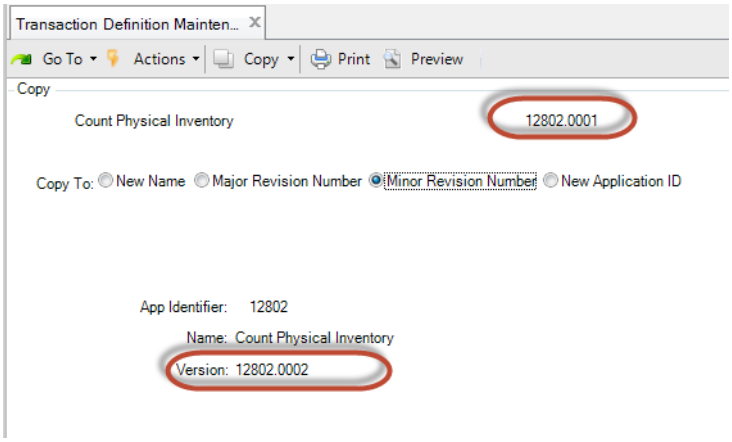


- **Minor Revision Number.** When this option is selected, the minor revision portion of the version number (the last two digits to the right of the decimal) is incremented by one. The major revision portion of the version number (the first two digits to the right of the decimal) is set to zero.

  When the new transaction is created:

  - The transaction name for the new transaction is the same as the source transaction.
  - All notes and history records are copied to the new transaction.
  - The application identifier is the same as the source transaction.

Fig. 3.39
Copy Definition - Minor Revision Number

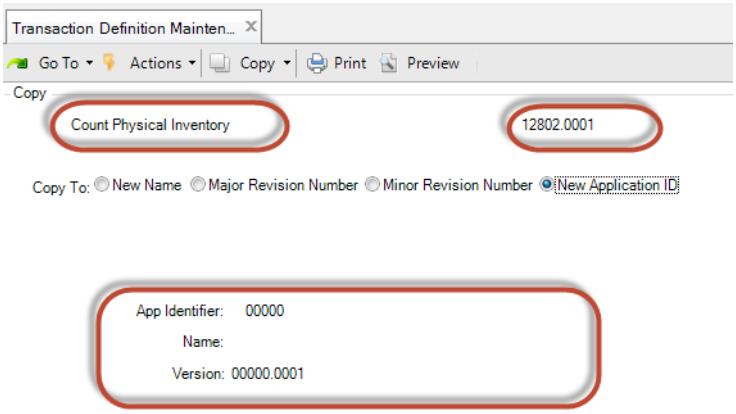


- **New Application ID.** When this option is selected, the user enters a new application identifier and new name for the transaction.

  When the new transaction is created:

  - The version is set by the system. The version is made up by the application identifier (the four digits to the left of the decimal) and the major and minor revision numbers.
  - The major revision number (the first two digits to the right of the decimal) is set to zero.
  - The minor revision number (the last two digits to the right of the decimal) is set to one (01).

Fig. 3.40
Copy Definition - New Application ID



Disable/Enable Transaction Navigation Keys

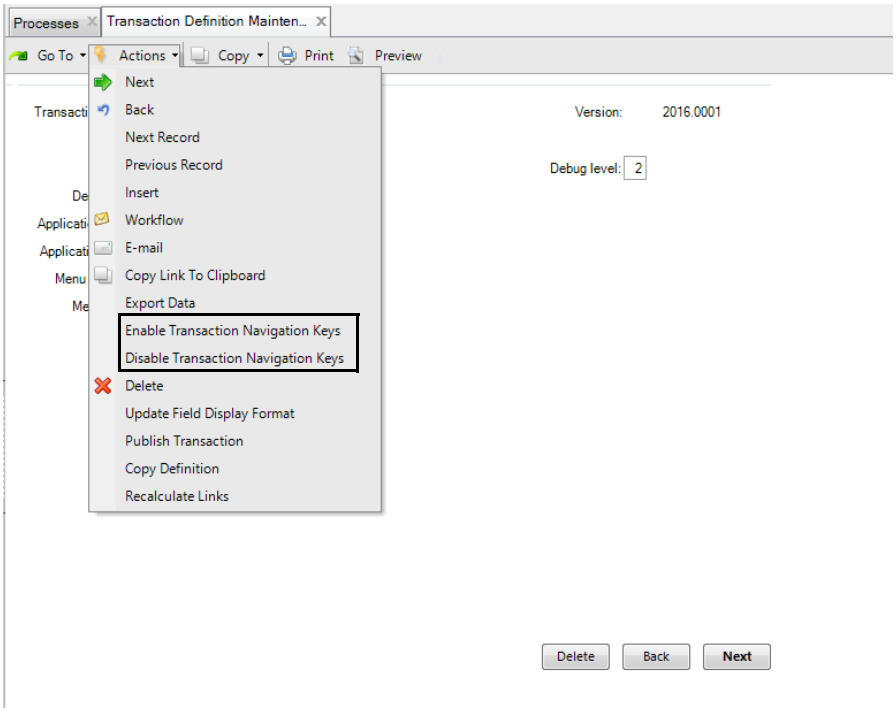
The Enable/Disable Transaction Keys actions are available from the header frame in Transaction Definition Maintenance (6.16).

- **Disable Transaction Navigation Keys.** Sets the Back Key Available, Done Key Available, and Abort Key Available settings to No.
- **Enable Transaction Navigation Keys.** Sets the Back Key Available, Done Key Available, and Abort Key Available settings to Yes.

For more information about the Back/Done/Abort Key Available fields, see “Maintain Dataset Buffer Field Information” on page 49.

Note These actions are only available in .NET from within the Debug Group (setting the debug and the debug level).

Fig. 3.41
Disable/Enable Transaction Navigation Keys



Chapter 4

Create the Field Device Menu

This chapter discusses the following topics:

Overview

Once you perform basic setup and build the transactions, you are ready to build the mobile field device structure for the transactions.

To do this, you:

1. Create categories for the mobile field device transactions.
2. Define a task type for RF device users that is based on a QAD EE program for the task.
3. Create a cross-reference between tasks linked to a category and process with actions (steps) that can be executed when the system executes a transaction for the task.
4. Gather data by running the actual mobile field device menu and transactions from within the Data Collection module.

Create Process Categories

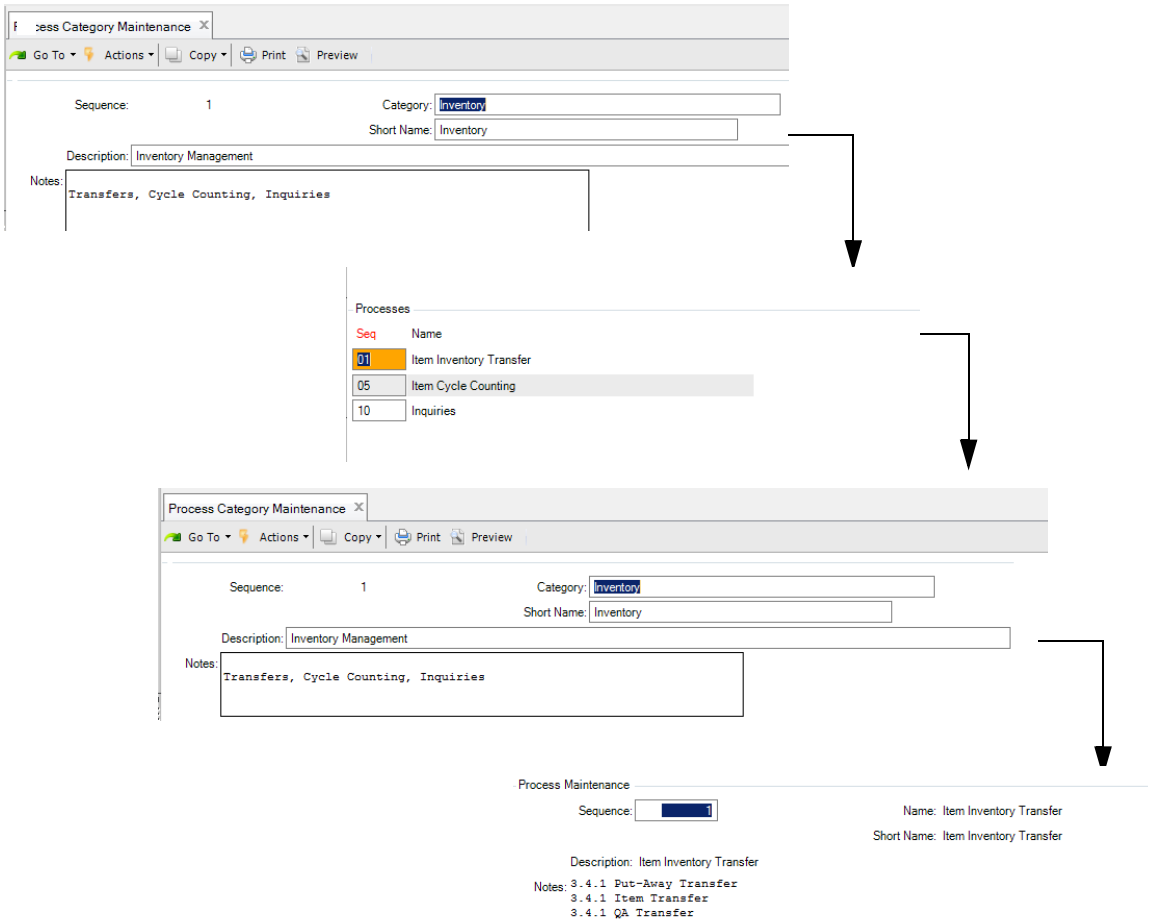
Use Process Category Maintenance (6.12) to define categories that group functionality for the mobile field device transactions. The categories define the menu structure for the RF device. The system assigns a sequence number to categories and the processes to indicate the order in which they are presented to the user for selection from the mobile device. The categories are user definable as well as being supplied as seed data with the product. Categories help RF users quickly find the processes and transactions to which their work pertains and shortens their access time.

Navigation

Begin by entering the sequence that you want when displaying the category; then, enter the process category name. Enter a meaningful name so that users who use the transactions within the category can easily access the category, such as inventory, cycle count, or shipping. Enter a short name for the category. Optionally, enter a description and any notes for the work category.

Press Next to enter processes—which become a submenu for the process category. Enter the sequence number for the first process for the category by name, short name, a description, and any notes for the process. Continue adding sequence numbers and processes for the category.

Fig. 4.1
Process Category Maintenance (6.12)



Sequence. Enter the sequence for the work category.

The sequence must be unique and greater than 0 (zero) for the Parent ID, and greater than 0 for the program detail ID or procedure or function.

Category. Enter the name of the work category. This field cannot be blank.

Short Name. Enter a short name for the work category. This field cannot be blank.

Description/Notes. Optionally, enter a description of and any notes for the work category.

Sequence (Process). Enter a sequence number in which the process displays on the mobile device or RF device.

Name. Enter the process name.

Short Name. Enter a short name for the process.

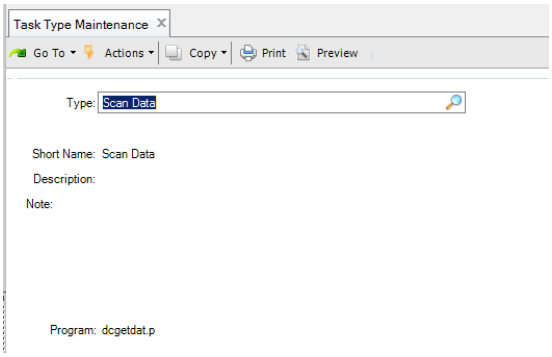
Description/Notes. Optionally, enter a description and any notes.

Create Tasks

Use Task Type Maintenance to define a task type for RF device users that is based on a QAD EE program for the task; for example, create a task type for scanning data.

You enter a task type name, the short name as it displays on the UI, reports, and other areas, then, optionally enter a description, followed by the Progress program name.

Fig. 4.2
Task Type Maintenance (6.13)



Type. Enter the type of task; for example, enter ScanData.

Short Name. Enter a short name for the task; for example enter ScData.

Description. Optionally, enter a description of the task type.

Note. Optionally, enter any notes about the task type.

Program. Enter the QAD EE Progress program name (.p name) for the task type.

Create a Task and Category/Process Cross-Reference

Use Process/Task/Definition Xref (6.17) to create a cross-reference between tasks linked to a category and process with actions—the tasks— that can be executed when the system executes a transaction for the task. Once you connect the task and category/processes, you can go on to attach the transaction to the task-category/process connection.

For example, you can define a shipping category with a QAD Truck Load process associated with it; then, cross-reference a scan data task with the work category; and finally, define the steps that determine when users scan data during shipping, such as Truck Load Base Step 1, Truck Load Base Step 2, and so on. You define tasks in DC Task Type Maintenance.

Navigation

1. Enter a valid process category as set up in Process Category Maintenance.

   The system displays the process associated with the category.

2. Select a valid process and click next to enter the Task Type and its sequence.

3. Enter the sequence number for the category task type, and then enter or select the task type.

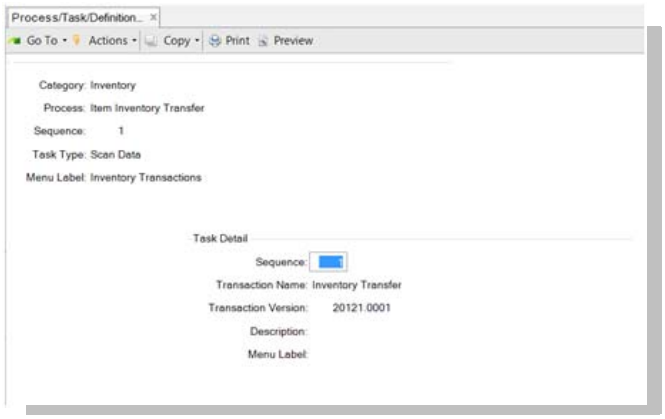
4 Enter the Task Type menu label.

When left blank, the system displays the task type as the label. Click Next to see the Task Detail frame.

5 Click Next to enter task detail data, including the sequence number, and the transaction name and version. Optionally, enter a description for the task detail.

6 Enter the menu label.

Fig. 4.3
Process/Task/Definition XRef (6.17)



Category. Specify a valid work category, or use the drop-down lookup browse to select a work category.

Sequence (task type). Enter the sequence number for the task type.

Process. These fields are view only and default from DC Work Category Maintenance setup.

Task Type. Enter a valid task type to associate with this process.

Menu Label (task type). Enter the menu label of the task type.

Sequence (action). Enter the sequence number for the action.

Transaction Name. Specify the action—the task—the user is to take for this process or use the lookup browse to select an action definition. These action definitions are set up using Transaction Definition Maintenance (6.16); see “Create Transactions” on page 38.

Action Version. Specify the action version.

Description. Optionally, enter a description of the action.

Menu Label. Enter the label for the menu that displays on the mobile field device. When you leave this field blank, but you have a description, the description displays on the Mobile Field Device menu. When Description is also left blank, the Action Definition name displays on the mobile field device menu.

Gather Data

You can run the menu with transactions that you built for the mobile field device directly from within the Data Collection module; however, you can only run the mobile field device menu and transactions in character mode.

You can test each transaction, gather data, then ensure that the data is valid and is reported correctly from within QAD EE.

To do this, you use the Gather Data (6.1) program. The system displays a menu in Gather Data that is a simulation of what the mobile field device user sees.

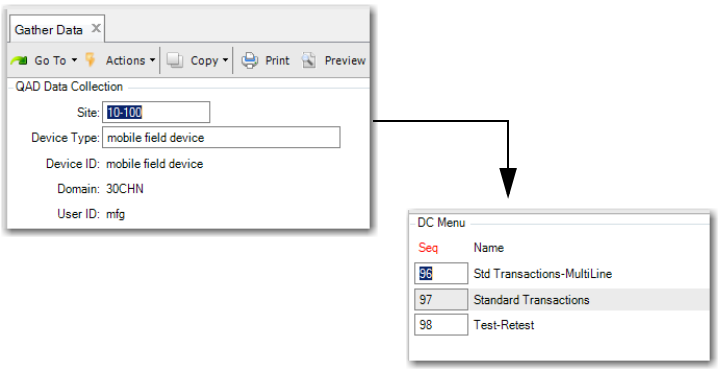
You can review your menu, program, category, and overall transaction structure. You can run the transactions, commit the transactions, send the collected data to the system, and test the received data.

Logging in to the Data Collection System

In the header, start by entering the site and device type. When you log in to the Data Collection system, the login site and device type default to the last settings for the user. You can optionally specify new values.

You enter a user ID and password that is longer than the field display. Although the fields display a maximum of eight characters, you can scroll to view up to 80 characters. The login recognizes new and soon-to-expire passwords.

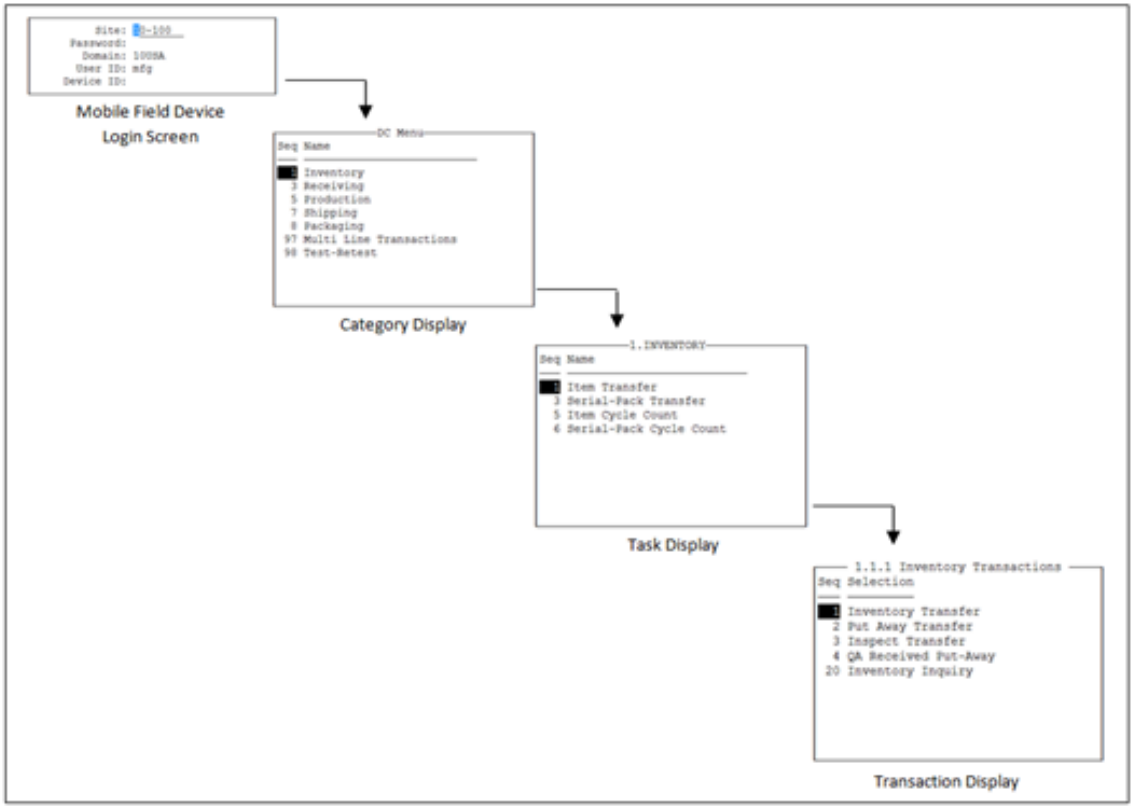
Fig. 4.4
Gather Data, Header



The following graphic depicts the character version of Gather Data (6.1). It shows the initial mobile field device login screen, the display of categories, tasks, then transactions. Based on the device ID, the mobile field device screen layout is rendered.

Once you run the transactions, use QAD standard reports, browses, and browse collections to review the Data Collection.

Fig. 4.5
Gather Data, Example Setup



Product Information Resources

QAD offers a number of online resources to help you get more information about using QAD products.

QAD Forums (community.qad.com)
Ask questions and share information with other members of the user community, including QAD experts.

QAD Knowledgebase (knowledgebase.qad.com)*
Search for answers, tips, or solutions related to any QAD product or topic.

QAD Document Library (documentlibrary.qad.com)
Get browser-based access to user guides, release notes, training guides, and so on; use powerful search features to find the document you want, then read online, or download and print PDF.

QAD Learning Center (learning.qad.com)*
Visit QAD's one-stop destination for all courses and training materials.

*Log-in required