

**precisely**

**ACR/Instream**

IM Tutorial



# Notices

## Trademarks

Infogix, the Infogix logo, ACR, ACR/Detail, ACR/Summary, ACR/Workbench, ACR/Connector, Infogix Assure, Infogix Insight, ACR/Instream, ACR/File, Infogix ER, Infogix Perceive, Data3Sixty, and Data360 are registered trademarks of Precisely. Data3Sixty Analyze, Data3Sixty Govern, Data3Sixty DQ+, Data360 Analyze, Data360 Govern and Data360 DQ+ are trademarks of Precisely. Any other trademarks or registered trademarks are the property of their respective owners.



1700 District Ave Ste 300  
Burlington MA 01803-5231  
USA

[www.precisely.com](http://www.precisely.com)

Copyright 2014, 2024 Precisely



## Table of Contents

<b>Notices .....</b>	<b>1</b>
Trademarks.....	1
Who Should Use this Guide?.....	4
What's Covered in the Tutorials? .....	4
Other Sources of Information.....	5
<b>Contact Us.....</b>	<b>6</b>
Community.....	6
Document Conventions .....	7
ACR/Instream Knowledge Base .....	7
Controls Design Concepts .....	9
Key ACR/Instream Terms .....	9
ACR/Instream User Interface .....	12
Control Components.....	13
What Is the ACR/Instream Domain?.....	16
Tutorial 1 .....	17
Value Validation .....	17
Summary of the CLAIMS Item Group .....	41
Tutorial 2.....	43
Loading and Testing Rules.....	43
Tutorial 3.....	55
Transaction Lifecycle Tracking .....	55
Tutorial 4 .....	69
Validating the Process Flow .....	69
Glossary .....	81

# Introduction

This guide is your starting point for learning how to write application controls, in the form of rules, to implement ACR/Instream.

## Who Should Use this Guide?

If you are responsible for any task in any implementation phase, and you have ACR/Instream installed on Windows, you can benefit from using these tutorials. For rules writers and programmers responsible for the design phase, these tutorials provide valuable training in controls development.

If you have not done so already, please read the *ACR/Instream Implementation Guide*. The implementation guide introduces you to the design phase and how it fits in the overall implementation process.

## What's Covered in the Tutorials?

This guide introduces you to a basic controls design and implementation, including the following:

- Controls design concepts and key terms
- Basic applications, such as balancing transactions and verifying system flow
- Basic control components, such as item groups, elements, and actions
- Basic comparison rule design, including conditionally activating rules

---

## Other Sources of Information

The table below describes what's in each chapter.

Chapter	Contents
"Controls Design Concepts"	An overview of basic controls design concepts.
"Tutorial 1"	Instructions for value verification.
"Tutorial 2"	Instructions for loading and testing controls.
Tutorial 3"	Instructions for enhancing the rules from Tutorial 1 to perform transaction lifecycle tracking.
"Tutorial 4"	Instructions for enhancing the rules from tutorial 1 and tutorial 3 by validating the process flow.

## Other Sources of Information

The following table provides a complete list of ACR/Instream documents.

Consult this document:	For this type of information:
<i>ACR/Instream Implementation Guide</i>	How to implement ACR/Instream. This guide also includes specific steps for the analysis phase of the implementation. Use this guide before installing the ACR/Instream software.
<i>ACR/Instream Installation Guide for z/OS or ACR/Instream Installation Guide for Windows</i>	How to install the ACR/Instream software.
<i>ACR/Instream Controls Design Guide</i>	Detailed information about designing and writing controls using ACR/Instream Editor.
<i>ACR/Instream Programmer's Guide</i>	Information about transporting messages, the data dictionary, integrity message layouts, tuning, and testing. Administrative information.
<i>ACR/Instream Knowledge Base</i>	The Knowledge Base is a repository of all user information available for ACR/Instream.

## Contact Us

If you encounter any technical issues, we recommend that you visit the support portal at [support.infogix.com](https://support.infogix.com).

If your query has not been discussed previously, you can create a new topic and receive answers from our product experts.

Alternatively, you can log a support ticket:

1. Select **Sign in** from the top right corner of the screen.
2. If you have already registered, enter your **Email** and **Password**, then click the **Sign in** button. Or, if you are not a registered support portal user, click **Sign up**.
3. Once you have registered and signed in, select **Submit a request** from the top right corner of the screen.
4. Complete all fields, then click **Submit** at the bottom of the screen.

## Community

Our product is constantly evolving and input from you is highly valued. If you have any suggestions, please contact the product team by submitting a feature request on the [Community](#).

## Document Conventions

This guide uses the following conventions for menus and shortcuts:

- A > indicates a menu choice. For example, **File > New** describes selecting the **New** command from the **File** menu.
- A + (plus sign) specifies, when describing keyboard actions, to hold down two keys together. For example, **CTRL+END** describes holding down the control key and then pressing the **END** key.

## ACR/Instream Knowledge Base

The ACR/Instream Knowledge Base is an HTML-based repository of all user information for ACR/Instream, plus samples and examples for downloading. The Knowledge Base launches in your default browser and provides standard browser-based searching capabilities so you can easily locate what you need.

The following describes some of the contents:

- All ACR/Instream manuals are available in PDF format. You can view these online or download to your PC and print.
- All ACR/Instream help files that are accessible from the user interfaces are also accessible as compiled help files that you can download. For example, the help file for the ACR/Instream Editor can be downloaded independently of the user interface.
- Specialized guides, such as the following:
  - IIP Database Guide
  - ACR/Instream Timer Utility User's Guide
  - ACR/Instream System Console XML Messages User's Guide

## Introduction

---

### *ACR/Instream Knowledge Base*

- ACR/Instream Flat-File Reader User's Guide
- Implementation support provides instructions, samples, and other resources to assist you in implementing ACR/Instream.
- Jump start solutions include sample rules files that you can download to your PC and adapt for your application.
- FAQs about rules writing addresses common queries about using the ACR/Instream Editor to develop rules.
- FAQs about implementation addresses common queries about how to get ACR/Instream up and running.
- Sample message files can be downloaded and used immediately without coding.
- Troubleshooting helps solve common problems.

The Knowledge Base is available two ways. A CD-based version is shipped with the installation media. This CD launches the Knowledge Base as an HTML-based help system.

The Knowledge Base is also available on the Internet from the Infogix Customer Support web page. Contact Customer Support for information about obtaining access.

# Controls Design Concepts

This chapter introduces you to the controls design concepts that form the foundation for the tutorials. These concepts include the following:

- Key ACR/Instream terms
- ACR/Instream user interface
- Control components

As you perform the tutorials, you'll learn more about these concepts and how you can use them for assuring the integrity of your target applications.

## Key ACR/Instream Terms

The following terms, which are discussed in this section, are key to your understanding of designing controls for your applications:

- Business objects
- APIs and middleware
- Integrity messages
- ACR/Instream Editor, ACR/Instream Player, and the ACR/Instream Data Dictionary

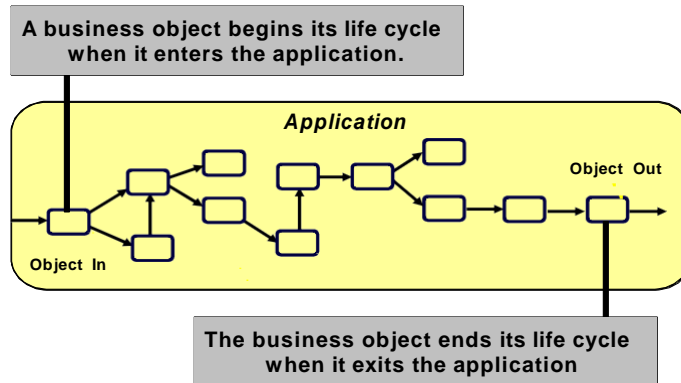
## Controls Design Concepts

### Key ACR/Instream Terms

---

## Business Objects

A business object is a collection of related data items which represents something in the real world, such as an insurance claim or a bank transaction, and which passes through an application system more or less intact. Each occurrence of any business object has a unique identity. Many ACR/Instream users refer to business objects as transactions.

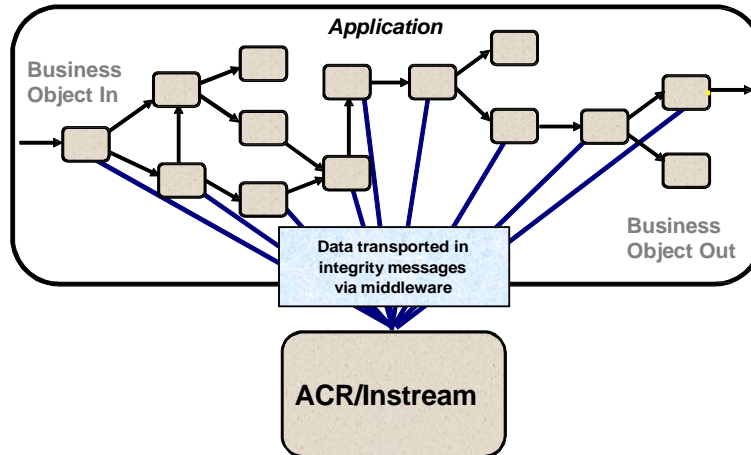


You can think of a business object as having a life cycle, which begins when it enters an application. During the life cycle of the object, it is acted upon by one or more software processes. Each process changes the state of the object. It is these changes in state that are tested by ACR/Instream to detect information integrity errors. The life cycle ends when it exits the application.

## APIs

APIs are software routines that capture information about business objects in your application and send the information to ACR/Instream. The programmer responsible for your implementation will insert the APIs in your target application. You'll need to choose, along with the programmer, at which point in the process you want data extracted.

For the tutorials, you do not need APIs. You'll learn how to simulate the APIs so you can develop your controls independently of your application.



### Integrity Messages

An integrity message is a set of data sent from the API in the target application to ACR/Instream. It is the sole means of communication between the target application and ACR/Instream. An integrity message performs the following functions:

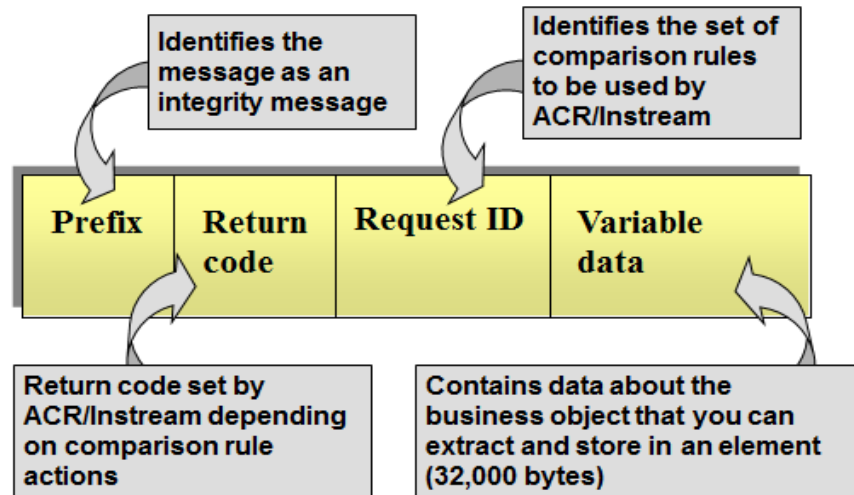
- Contains information that identifies itself as a valid integrity message
- Provides data about the business object that can be used in the controls
- Identifies the set of controls to apply to the message

## Controls Design Concepts

### ACR/Instream User Interface

---

Integrity messages can also be sent back to the target application, providing feedback. The diagram below shows the structure of an integrity message.



Development of the integrity messages from the target application is the responsibility of a programmer. Your focus, as a rules writer, is to write controls that extract the data from the message and apply comparison rules that evaluate the data and verify the application's integrity.

For the tutorials, integrity messages are provided that you can use to test your controls.

## ACR/Instream User Interface

While performing the tutorials, you'll use the following user interface components:

- ACR/Instream Editor
- ACR/Instream Player
- ACR/Instream Data Dictionary

### ACR/Instream Editor

The ACR/Instream Editor is a tool for rules writers—those individuals who write the controls that solve your business problems. With ACR/Instream Editor's user-friendly interface, you can create controls for your application that can then be tested.

### **ACR/Instream Player**

The ACR/Instream Player is a tool for assessing the proper functioning of your controls. The ACR/Instream Player reads an ACR/Instream message log file and sends the messages to a domain. The output values, return codes, and other data are displayed as the messages are executed.

By playing the log into ACR/Instream repeatedly, you can develop your ACR/Instream controls in an orderly, step-by-step fashion. For the tutorials, an integrity message log file is provided so you can use ACR/Instream Player to test the controls you create in the tutorials.

### **ACR/Instream Data Dictionary**

The ACR/Instream data dictionary forms a bridge between the integrity message, containing the application's data, and the creation of the controls using ACR/Instream Editor. The data dictionary contains technical information about layout definitions of the target application. A programmer familiar with the layout definitions uses the ACR/Instream Data Dictionary Editor to enter them into the data dictionary and to assign them easy-to-understand names.

This makes it possible for nontechnical users to create their own rules by selecting the names using a drop-down list. Rules writers do not need to know the exact position, length, and data type of the information. If your source data is an XML document, the rules writers do not need to know the node or attribute.

## **Control Components**

This section introduces you to some basic control components that you'll use in the tutorials:

- Items and item groups
- Extracted and calculated elements
- Comparison rules and actions
- Request definitions

Additional components are introduced throughout the tutorials. Also, see the *ACR/Instream Controls Design Guide* for a complete discussion of components.

## Controls Design Concepts

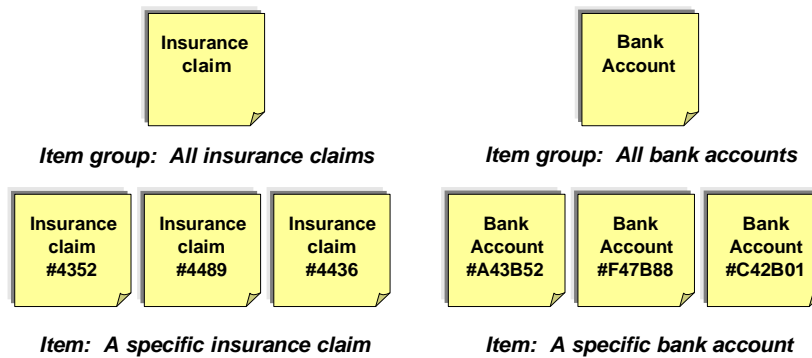
### Control Components

---

## Items and Item Groups

An item is a specific instance of a business object with a unique identifier. For example, an insurance claim could be considered a business object. Its specific claim number can uniquely identify it.

An item group is a template used by ACR/Instream to describe every individual member of that group. Define one item group for each business object you want to control. For example, an item group of *All insurance claims* could describe each insurance claim or an item group of *All bank accounts* could describe each bank account.



All ACR/Instream control components are associated with a particular item group. Therefore, the first task you perform (after accessing ACR/Instream) is to define the item group.

## Elements

An element is data associated with the business object that can be used in the controls. The tutorials focus on two types of elements: extracted and calculated.

An *extracted element* is taken directly from the integrity message sent from the application by the API. For example, the amount of an insurance claim could be extracted data.

A *calculated element* is created by a formula that uses other data, including extracted elements. For example, a counter that increments each time a process is complete is a calculated element.

## Comparison Rules and Actions

Comparison rules perform the checking of data from the target application to determine the integrity of the target item's information.

A comparison rule is a formula that represents the target item's state. For example, a comparison rule for an insurance claim could test transactions with the following formula:

$$\text{BEGINNING AMOUNT} \text{ EQ } \text{ACCEPTED AMOUNT} + \text{REJECTED AMOUNT}$$

The elements in a comparison rule can either be extracted from the integrity message or calculated by another formula.

After ACR/Instream performs the comparison, it can take an action that you define based on a TRUE or FALSE outcome. For example, ACR/Instream can:

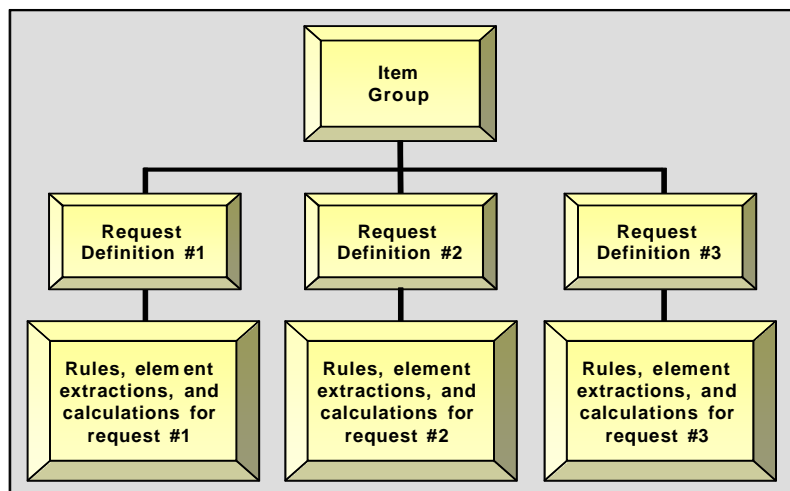
- Print a report
- Trigger actions based on the outcomes of the comparison
- Send an alert of an error to an operator
- Send a numerical response called a return code back to the application

Together, the comparison rules and actions form the foundation of your controls design. The tutorials will introduce you to multiple comparison rules and actions.

## Request Definitions

A request definition is a method to identify a set of controls to be used by ACR/Instream for processing an integrity message. When ACR/Instream receives an integrity message from an application, it includes the request definition ID, which identifies the specific request definition to apply.

An item group can have as many request definitions as necessary. Each request definition has its own set of element extractions, calculations, and rules.



## Controls Design Concepts

---

### *What Is the ACR/Instream Domain?*

For example, the tutorials use an item group for insurance claims and request definitions for:

- Registered claims
- Accepted claims
- Rejected claims
- Approved (completed) claims

In the tutorials, you'll create multiple request definitions that check the integrity of an insurance claim throughout its life cycle. Each request definition will be appropriate for the current state of the business object. For the sample application in the tutorials, a request definition will be required for each process that the claim completes.

## What Is the ACR/Instream Domain?

The *ACR/Instream domain* refers collectively to ACR/Instream's internal servers, definition files, and the *ACR/Instream domain manager*.

The ACR/Instream domain manager manages incoming and outgoing integrity messages for the ACR/Instream servers. These messages come from the APIs inserted into the target application. The APIs in your applications can be on any platform to send messages to the ACR/Instream domain manager.

# Tutorial 1

Controls design in ACR/Instream can be simple or complex. Regardless of how complicated your controls are, they still are usually constructed along three principles:

- Extract the data you need from the integrity message, sent from the application
- Evaluate the data from the application by applying comparison rules
- Take action based on the evaluation

In this first tutorial, you will learn how to construct your first rules using the above three principles. After you complete this tutorial, you'll be able to:

- Access ACR/Instream Editor
- Define item groups and elements
- Create request definitions
- Define how to extract element values from integrity messages
- Define comparison rules and the actions that ACR/Instream takes as a result of those rules

## Value Validation

Value validation ensures that inputs and outputs of a specific process conform to a defined relationship.

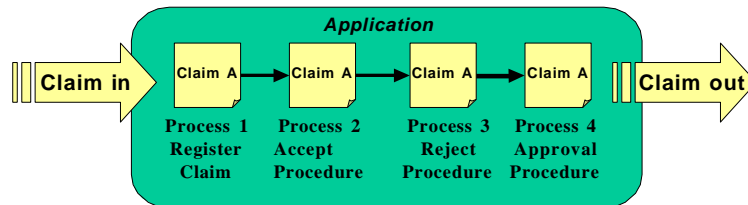
A process has some type of logical relationship identified between the inputs that enter the process and the outputs generated from the process. ACR/Instream monitors the process and verifies that the inputs and outputs conform to this relationship.

# Tutorial 1

## Value Validation

### Sample Application

This sample application creates rules that verify claims are processed correctly as they pass through a claims processing application. The problem is that sometimes transactions are not properly posted to the claim. If a claim is processed correctly, the amount rejected plus the amount accepted will equal the beginning amount of the claim.



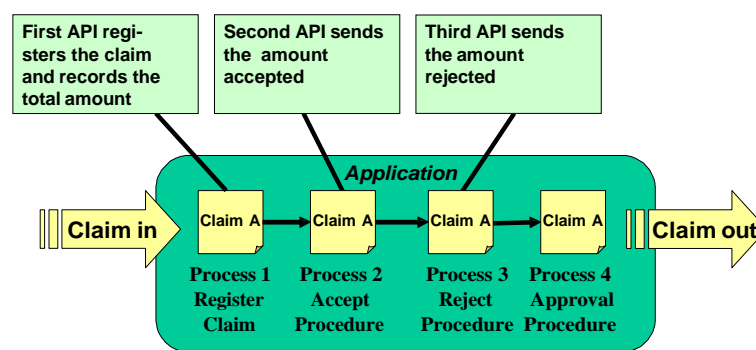
Each claim is processed multiple times throughout its life cycle within the application. Occasionally transactions are not processed properly for a claim.

A comparison rule is needed to verify this correct processing and then take appropriate action:

- If the amounts do balance, data about the item is erased from ACR/Instream's memory.
- If the amounts do not balance, a message is sent to the application administrator and a numerical response called a return code is sent back to the application.

### APIs for the Tutorial

For this tutorial, you can assume three APIs are inserted: one for each of the first three processes. In a real application, these APIs continuously send integrity messages to ACR/Instream. An API for the fourth process will be added in Tutorial 3.



### **Integrity Messages for the Tutorial**

For the tutorial, sample integrity messages are provided that simulate the functioning of the APIs. These sample integrity messages were provided on the installation media. The following shows a sample integrity message:

```
UXPIIEU 0400025500000000000000000000 S INTEGRITY CHECK      200006111600125106  
REGISTER THE CLAIM                1234567 0045399 0000000    08
```

In real life, this message is one continuous record. For this manual, the message is split to fit the page. In this message, you can locate the following elements:

- The first line is the prefix area, which contains data that ACR/Instream needs to process the message. INTEGRITY CHECK is the function that ACR/Instream is to perform.
- REGISTER THE CLAIM is the name of the request definition to be executed.
- The number 1234567 represents the unique identifier of the claim.
- The number 0045399 is the amount of the claim when it is registered.

### **Data Dictionary for the Tutorial**

A sample data dictionary is provided for the tutorial. This sample data dictionary contains descriptions of the integrity message layouts for the sample application. As you complete the tutorial, you will be able to choose elements in the integrity message using a drop-down list.

ACR/Instream also supports XML-based data dictionaries. See the *ACR/Instream Controls Design Guide* for more information.

## **Concepts for this Tutorial**

This section introduces concepts you'll apply in this tutorial: multiple claim processing and return codes.

### **Multiple Claim Processing**

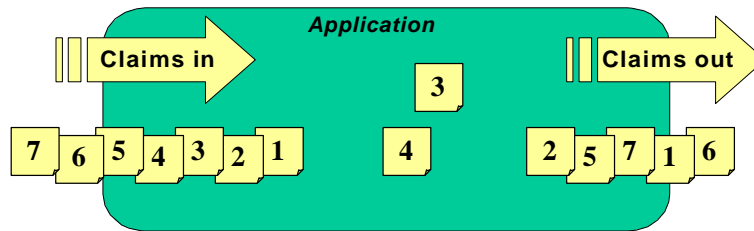
Most systems like this sample application process several claims simultaneously. One or more claims can be at any step in the process at any given time.

For example, the first API may report the entry of 24 claims into the system before the first claim is reported as leaving the system at the last API point. In addition, it is possible that claims 1, 2, 3, 4, 5, and 6 enter the system and

# Tutorial 1

## Value Validation

leave the system in a completely different order—perhaps 6, 1, 3, 5, and 2. This is the nature of transactional systems. ACR/Instream tracks all transactions regardless of order.



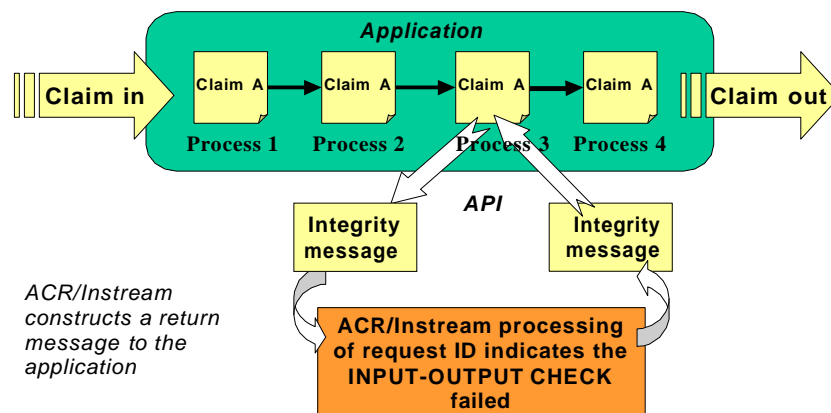
From the standpoint of controls design, you do not need to worry about the quantity or order of transactions. You can write your rules as if each claim were the only one being processed by the system, or more accurately, as if the system processes one claim at a time, completing each claim before starting a new claim. ACR/Instream takes care of the fact that 10, 100, or even 10,000 claims might be in the system at the same time.

The integrity messages supplied for this tutorial simulate the receipt of data about multiple claims in random order. When you test your new rules in Tutorial 2, you will see messages about the various claims.

### Return Codes

A return code is a number that you can use to report the outcome of the rule evaluation to the application. The code is returned to the API that sent the integrity message.

For example, if the rule evaluation indicates a problem in the application, ACR/Instream can send the application a return code that indicates the processing should stop until the problem is resolved.



*Work with the control design programmer to determine what codes to use and how the application responds to the codes*

Not all rules use a return code. When you design your controls, you'll determine if it's necessary to respond to the target application program. This decision is made in Design Step 4, *Determine Application Program Response*.

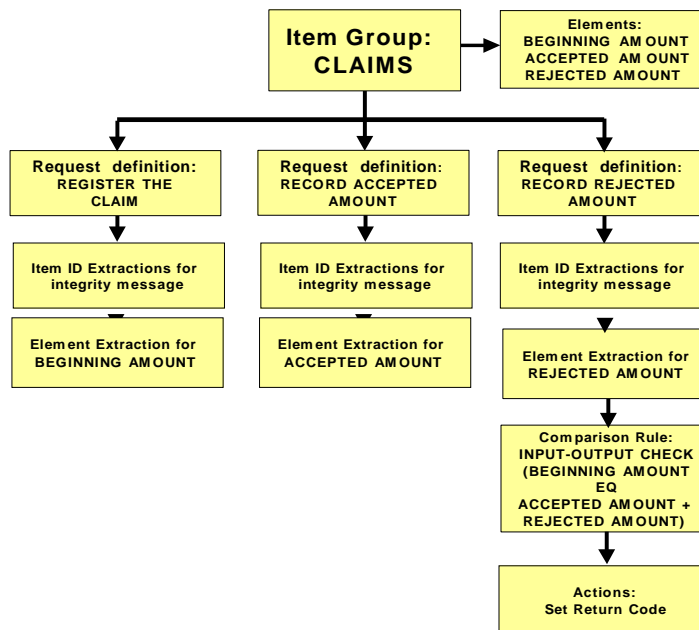
## Processing Strategy

This first tutorial uses a basic strategy of comparing the amount of the claim at the beginning of the application to the amount of the claim at the end of the third process:

BEGINNING AMOUNT EQ REJECTED AMOUNT + ACCEPTED AMOUNT

If this comparison has a result of FALSE, a return code is sent back to the application.

To achieve this strategy, you'll define an item group named CLAIMS. In the CLAIMS item group, you'll define the elements, request definitions, rules, and actions to achieve the processing strategy. The following diagram shows the structure of the item group you will create in this tutorial.



## Tutorial 1

---

### *Value Validation*

## Tasks for Tutorial 1

For this tutorial, sample integrity messages were installed with ACR/Instream that you can use to test your rules at the end of the tutorial. These integrity messages simulate the integrity messages that ACR/Instream would receive from the target application.

In this tutorial, you will perform the following tasks:

1. Copy all sample tutorial files to the domain folder and load the data dictionary.
2. Access ACR/Instream Editor and create the CLAIMS item group.
3. Add elements to the CLAIMS item group.
4. Create the request definition to register the claim.
5. Define the item ID extraction.
6. Define the element extraction.
7. Create the RECORD ACCEPTED AMOUNT request definition.
8. Create the RECORD REJECTED AMOUNT request definition.
9. Define the rule for the RECORD REJECTED AMOUNT request definition.
10. Define the actions for the comparison rule.
11. Save the rules file.

Keep the following in mind:

- Complete the tasks in order.
- You can stop at any time; be sure to save your work.
- Allow one to two hours to complete the tutorial.
- For a list of the complete item group contents, see [“Summary of the CLAIMS Item Group”](#)

### **Task 1: Copy Sample Tutorial Files and Load the Data Dictionary**

Ordinarily, loading the data dictionary into the ACR/Instream domain is the responsibility of the data dictionary programmer. After developing the data dictionary, the programmer loads it so you can select items from a drop-down list.

However, anyone can load the data dictionary file for the tutorials using the steps below.

---

**Note:** If you clicked the **Start ACR Instream Domain** icon, be sure to click the **Stop ACR Instream Domain** icon before continuing.

---

1. Locate the sample tutorial files on the ACR/Instream documentation CD and copy them to your domain folder. Copy the following files:

- Tutorial.imd (the data dictionary)
- Tutorial2.ieu (sample message file to use in tutorial 2)
- Tutorial3.ieu (sample message file to use in tutorial 3)
- Tutorial4.ieu (sample message file to use in tutorial 4)

The default location for the domain is the following:

```
c:\Program Files (x86)\Infogix\ACR Instream\domain
```

If your site installed ACR/Instream in a different location, you can use Windows Explorer to search for the command *run138l.bat*. The folder in which *run138l.bat* resides is the ACR/Instream domain folder.

---

**Note:** If you cannot locate the sample files, contact Infogix Customer Support.

---

2. Access an MS-DOS command prompt window. To do this on most Windows PCs, click **Start > Programs > Accessories**, and then click **Command Prompt**.
3. Navigate to your ACR/Instream domain folder.
4. Enter the following command at the MS-DOS prompt in the domain subdirectory:

```
runl38l tutorial.imd
```

Notice that the command uses a lowercase L as the last character. The L indicates the command is to be used for local mode.

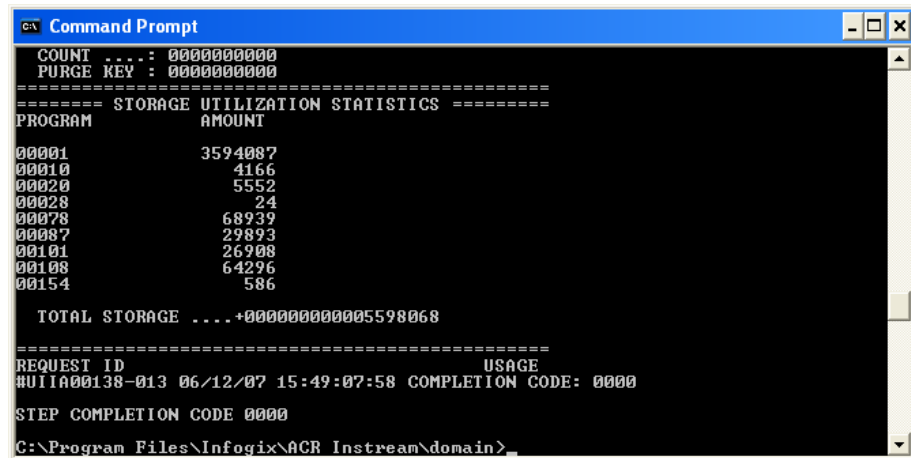
If tutorial.imd is not located in the same folder as the runl38l command, you need to specify the drive and path using the following command:

```
runl38l [drive:][path] tutorial.imd
```

Local mode applies when the ACR/Instream domain is located on your PC and the domain is not started. In Tutorial 2, you'll learn more about local and remote mode. For now, the instructions assume the ACR/Instream domain is installed on your PC but not started.

## Tutorial 1

### Value Validation



```
CA Command Prompt
COUNT ....: 0000000000
PURGE KEY : 0000000000
=====
===== STORAGE UTILIZATION STATISTICS =====
PROGRAM          AMOUNT
00001             3594087
00010              4166
00020              5552
00028               24
00078             68939
00087             29893
00101             26908
00108             64296
00154               586

TOTAL STORAGE ...+00000000000598068
=====
REQUEST ID          USAGE
#UIIA00138-013 06/12/07 15:49:07:58 COMPLETION CODE: 0000
STEP COMPLETION CODE 0000
C:\Program Files\Infogix\ACR Instream\domain>
```

5. Verify your completion code is all zeroes, indicating your rules have no errors. If you have an error, check the path that you specified for tutorial.imd. A path error is the most common type of error.
6. Open the output file that was generated by the run138l command. The default name of this file is *dispout.rpt*. You can use any text editor, such as WordPad, to review the output file. If your run138l command executed correctly, this file shows statements that look like the following:

```
UXPIIEU 04000333000000000000000000 S UPDATE DD TABLE
2000060616492320+0000
```

```
WRITE RETURN CODE 0000 LOGICAL ID 00000000
```

If you receive a message that looks like the following, then see your ACR/Instream programmer for assistance.

```
IIATABLE02DD      0100000000tutorial      claimnum
0000000000051     0000
#UIIA00138-021 DATA DICTIONARY POS=0 AND DELIM MISSING
```

7. Exit MS-DOS and return to the Windows desktop.

## Task 2: Access ACR/Instream Editor and Create the CLAIMS Item Group

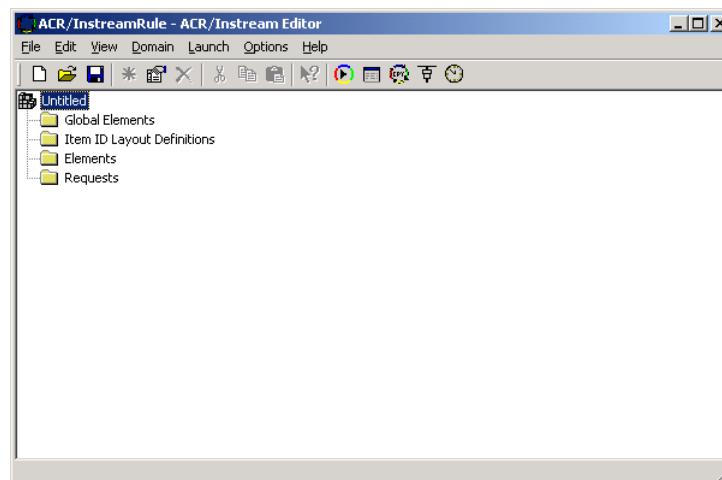
The item group you define in this task is a template that describes all claims that the application processes. For this tutorial, you define an item group that describes all insurance claims.



Item group: All insurance claims

If ACR/Instream Editor is not currently installed on your PC, please see the *ACR/Instream Installation Guide* before starting this tutorial.

1. Open the ACR/Instream Application folder and double-click **ACR Instream Editor** to open the ACR/Instream Editor main window. Or, click **Start**, then select **Programs > Infogix > ACR Instream > ACR Instream Editor**.



The default item group, labeled *Untitled*, is highlighted. The icon for an item group is shown at left.

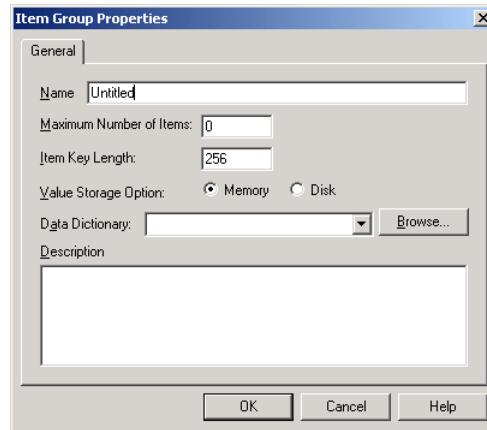
Be sure the name of the item group, *Untitled*, is selected.

## Tutorial 1

### Value Validation

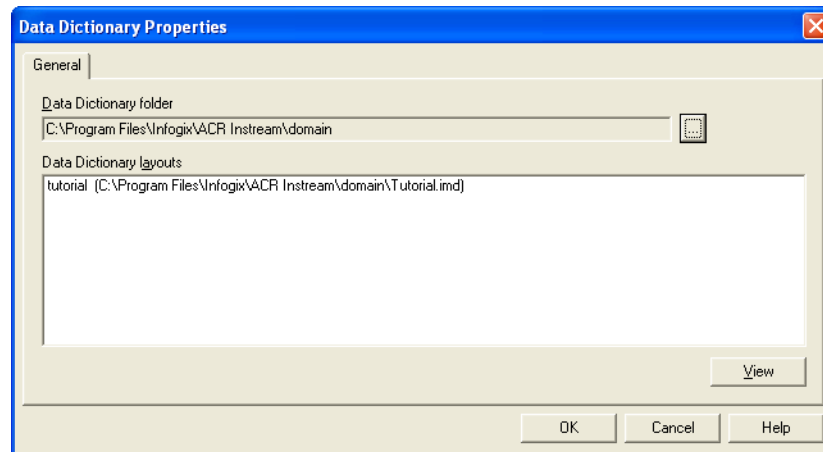
---

2. Select **Edit > Properties** to open the **Item Group Properties** dialog box. Or, double-click **Untitled**.



3. Enter **CLAIMS** in the **Name** box to identify your new item group.  
In these tutorials, the names of the components are in uppercase. This is not a requirement—you can use any alphanumeric combination up to 40 characters. You cannot start any name with a + (plus), - (minus), or ' (apostrophe).
4. Enter **20** in the **Maximum number of items** box.  
The *Maximum number of items* defines the initially allocated space for all the items in an item group. To define, estimate the number of items in the process at any given time. For this example, the number of items is the number of unique claims being processed at one time.  
ACR/Instream uses this number to allocate space to hold the number of items you specify under its control simultaneously. If you underestimate the number of items that need to be controlled simultaneously, ACR/Instream automatically increases the space by a predetermined percentage. It will do this as soon as the original number you specify here is reached. However, ACR/Instream must do this dynamically while your application is in process and further processing will wait until the new allocation is complete.  
If the new space subsequently fills, ACR/Instream increases it again. This process repeats each time the space fills.  
Use the default Item Key Length of 256 and Value Storage Option of Memory.

- Click **Browse** to open the **Data Dictionary Properties** dialog box.



- Review your dialog box to verify that *tutorial* is listed in the **Data Dictionary layouts** list (as shown above).

If it is not, click  to open the **Browse for Folder** dialog box and navigate to the following folder and click **OK**:

```
c:\Program Files\Infogix\ACR Instream\Domain
```

The file named *tutorial.imd* should now be listed under **Data Dictionary** layouts.

- Select **Tutorial** in the list box and click **OK** to return to the **Item Group Properties** dialog box.
- Select **Tutorial** from the **Data Dictionary** drop-down list.
- Enter **Balancing amounts for insurance claims** in the **Description** box and click **OK**.

The Item ID Layout Definitions folder and the Global Element folders are not used in these tutorials.

### Task 3: Add Elements to the CLAIMS Item Group

Elements are data that are either extracted from the integrity message sent by the APIs or calculated using other means. For the sample application in this first tutorial, you need three elements defined:


- Amount at beginning
- Accepted amount
- Rejected amount

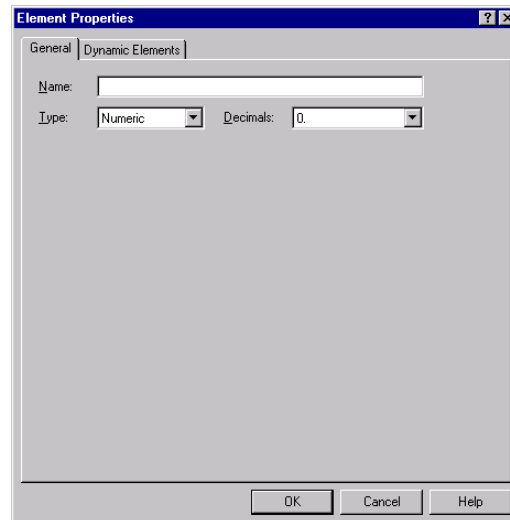
## Tutorial 1

### Value Validation


---

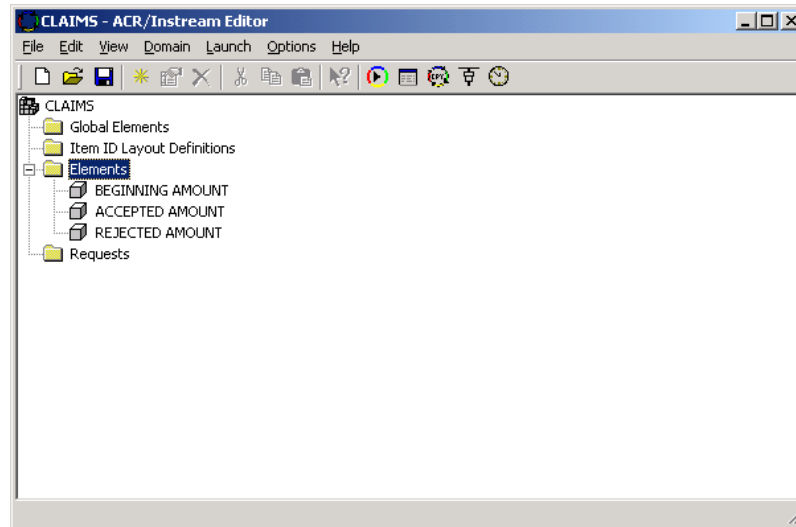
The ACR/Instream Editor main window must be showing before you continue.

1. Select the Elements folder and then select **Edit > New** to open the **Element Properties** dialog box.  
As an alternative, you can click  or you can select a folder and then press **Insert** to open a properties dialog box to create any ACR/Instream component.



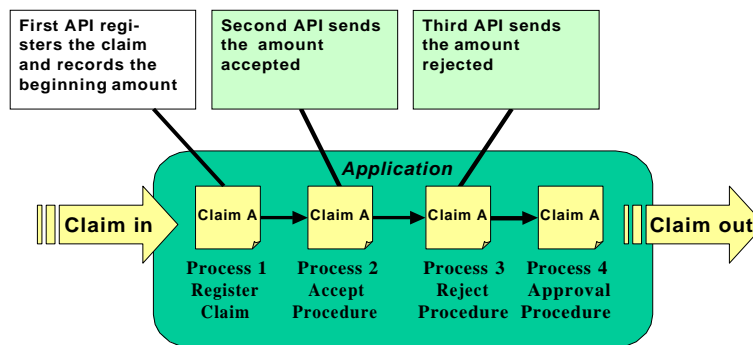
2. Complete the **Element Properties** dialog box for the first element. Use the data below and click **OK** when complete:  
Name: BEGINNING AMOUNT  
Type: Numeric  
Decimals: 0.
3. Repeat the above step for the second element, naming it ACCEPTED AMOUNT.
4. Repeat Step 2 for the third element, naming it REJECTED AMOUNT.

- Click the  symbol next to the Elements folder. Your ACR/Instream Editor main window should look like the following:



**Task 4: Create the Request Definition to Register the Claim**

When the integrity message is received from ACR/Instream, it includes the request definition ID that tells ACR/Instream which request definition (and subsequently which rules) to use.



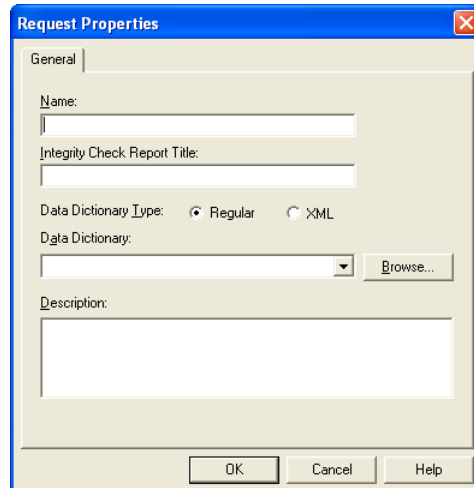
In this task, you will define a request definition for the integrity message to be received when the claim begins the application.


## Tutorial 1

### Value Validation

---

1. Select the Requests folder, then select **Edit > New** to open the **Request Properties** dialog box.



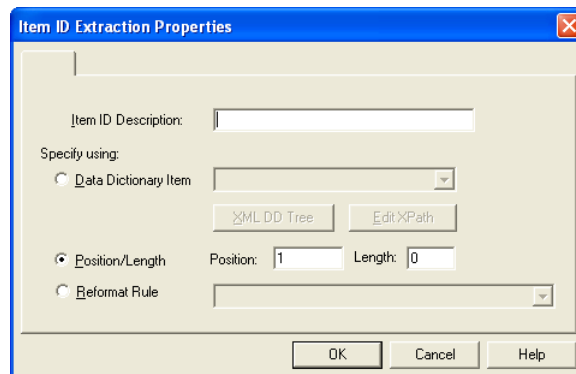
2. Enter **REGISTER THE CLAIM** in the **Name** box.  
Be sure to enter this name exactly as shown. This name must match the name in the integrity message exactly, including case. This is true not only for the tutorial, but for all integrity messages.
3. Select **tutorial** from the **Data Dictionary** list.
4. Enter **Register claim and record beginning amount** in the **Description** box and click **OK** to return to the ACR/Instream Editor main window.  
For now, leave the **Integrity Check Report Title** box blank.
5. Expand both the Requests folder and the new REGISTER THE CLAIM section under the Requests folder. To expand a section, click the  symbol.

The main window now shows the components that you can define for the new request definition.

### Task 5: Define the Item ID Extraction

Also included in the integrity message is the item ID, which identifies a specific item. In this sample application, the item ID is the insurance claim number. In this task, you'll define where the item ID is located in the integrity message.

1. Select the Item ID Extractions folder and select **Edit > New** to open the **Item ID Extraction Properties** dialog box.



2. Enter an item ID description of **Beginning Amount**. This description will be associated with the extracted value for the item ID.
3. Click **Data Dictionary** item, select **Claim number** from the drop-down list, and then click **OK**.

Notice that you do not have to specify the position or the length in the integrity message. That is because the **Claim number** item in the dictionary contains that data information.

This data dictionary list was created for this tutorial and shipped on the installation media. For your implementation, the controls design programmer will create the data dictionary.

## Tutorial 1

### Value Validation

---

#### Task 6: Define the Element Extraction

In Task 3, you defined the names of elements to be extracted from the integrity message. In this task, you define what element is to be extracted for the REGISTER THE CLAIM request definition.

1. Select the Element Extractions folder and select **Edit > New** to open the **Element Extraction Properties** dialog box.

Element Extraction Properties

General | Foreign Elements | Calendars

Element storage

Element Name: BEGINNING AMOUNT

Accum Operator: Replace

Extraction

Dictionary item: Claim amount at beginning

XML DD Tree Edit XPath

Position: 0 Length: 0

Format: Numeric

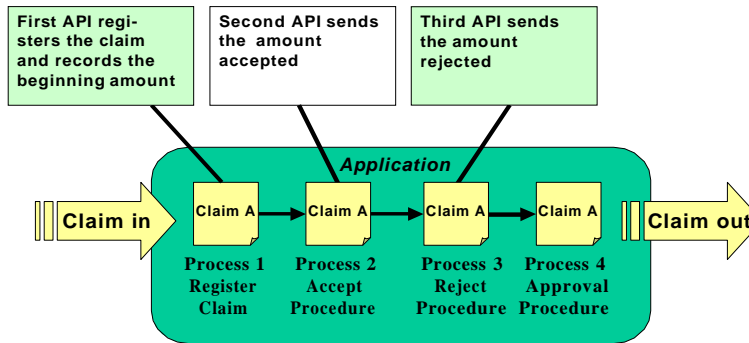
Note: If you change the Data Type and/or Attribute of an XML DD element, you must reselect that element from the XML DD Tree.

OK Cancel Help

2. Select BEGINNING AMOUNT from the **Element Name** list. Select the Accum Operator Add.
3. Select **Claim amount at beginning** from the Dictionary item list and click **OK**.

### Task 7: Create the RECORD ACCEPTED AMOUNT Request Definition

In this task you create a request definition to acquire data from the second API for process 2, the accept procedure.

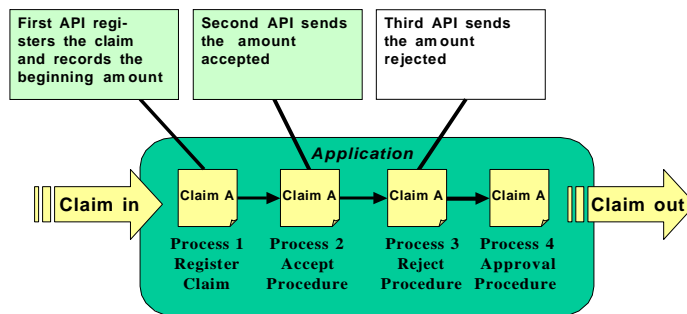


Note that many values will be the same as the first request definition. That's because the integrity message layouts for both API points are structured the same for this sample application. This method of integrity message layout design is recommended for your implementation as it simplifies the controls design process and API maintenance.

1. Select the Requests folder, then select **Edit > New** to open the **Request Properties** dialog box.
2. Enter **RECORD ACCEPTED AMOUNT** in the **Name** box. Again, be sure your request name matches exactly.
3. Select **Tutorial** from the Data Dictionary list.
4. Enter **Extract amount of claim accepted** in the **Description** box and click **OK**.
5. Expand the new RECORD ACCEPTED AMOUNT section.
6. Select the Item ID Extractions folder and then select **Edit > New** to open the **Item ID Extraction Properties** dialog box.
7. Enter an item ID description of **Accepted Amount**.
8. Click **Data Dictionary** item, select **Claim number** from the drop-down list, and then click **OK**.
9. Select the Element Extractions folder and then select **Edit > New** to open the **Element Extractions Properties** dialog box.
10. Select an element name of **ACCEPTED AMOUNT**. Select the Accum Operator Add.
11. Select a Data Dictionary item of **Approved amount** and click **OK**.

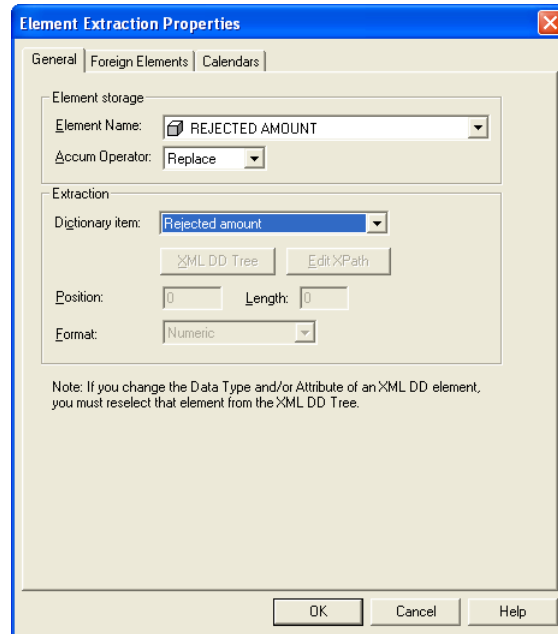
#### Task 8: Create the RECORD REJECTED AMOUNT Request Definition

In this task, you will define a request definition for the integrity message received when the claim finishes Process 3, the reject procedure. This is an *integrity checkpoint*, the point at which the integrity of the business object is verified before the object continues its life cycle.



1. Select the Requests folder, then select **Edit > New** to open the **Request Properties** dialog box.
2. Enter **RECORD REJECTED AMOUNT** in the **Name** box. Again, be sure your request name matches exactly so the tutorial will work correctly.
3. Select **Tutorial** from the **Data Dictionary** list.
4. Enter **Integrity Checkpoint** in the **Description** box and click **OK**.
5. Expand the new RECORD REJECTED AMOUNT section.
6. Select the Item ID Extractions folder and then select **Edit > New** to open the **Item ID Extraction Properties** dialog box.
7. Enter an item ID description of **Rejected Amount**.
8. Click **Data Dictionary** item, select **Claim number** from the drop-down list, and then click **OK**.
9. Select the Element Extractions folder and then select **Edit > New** to open the **Element Extractions Properties** dialog box.

10. Select **REJECTED AMOUNT** from the **Element Name** list. Select the Accum Operator Add.



11. Select **Rejected amount** from the **Dictionary item** list and click **OK**.

### Task 9: Define the Rule for the RECORD REJECTED AMOUNT Request Definition

In this task, you tell ACR/Instream what to do with the accumulated data by defining a comparison rule. The comparison rule performs the checking of element values to determine the integrity of the business object.

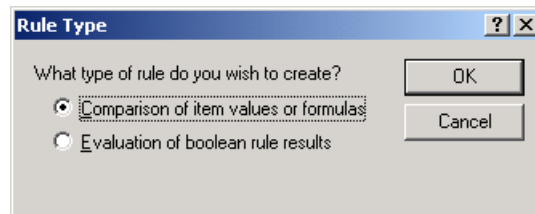
Note that a comparison rule is necessary only for the RECORD REJECTED AMOUNT request definition, since the first two request definitions are only for collecting data.

## Tutorial 1

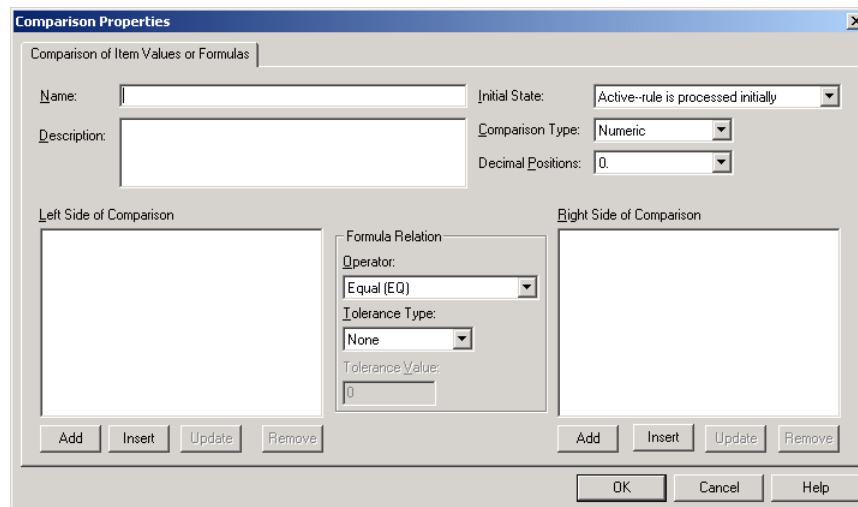
### Value Validation

Part of the comparison rule consists of a logic formula that compares stored values. Values can be elements or literal values. In this tutorial, you define a formula in which you want to know if the amount extracted in the beginning is the same as the total amount that is accepted and rejected.

1. Select the Comparisons folder under the RECORD REJECTED AMOUNT request definition and select **Edit > New** to open the **Rule Type** dialog box.



2. Select **Comparison of item values or formulas** and click **OK** to open the **Comparison Properties** dialog box.



The formula you are creating here is a straightforward expression of one element (BEGINNING AMOUNT) equaling the total of other elements (REJECTED AMOUNT + ACCEPTED AMOUNT).

3. Enter **INPUT-OUTPUT CHECK** in the **Name** box.
4. Enter **Compare beginning amount to current amount** in the description box.

Use the defaults for the **Comparison Properties**. The following lists the defaults:

Initial state: Active-rule is processed initially

Comparison Type: Numeric

Decimals Positions: 0.

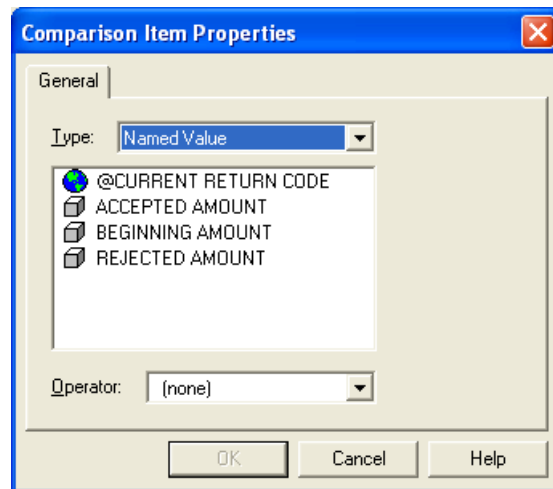
Operator: Equal (EQ)

Tolerance Type: None

Tolerance Value: 0 (this item is grayed out at this step)

You'll come back to the **defaults** and review them in a later step.

5. Click in the **Left Side of Comparison** pane to make it active.
6. Click **Add** to open the **Comparison Item Properties** dialog box.



In this dialog box are the elements you defined in earlier task. They are now available for selection.

7. Select **BEGINNING AMOUNT** and then click **OK**. Leave the default for the operator as **None**.
8. Click in the pane labeled **Right Side of Comparison** to make it active.
9. Click **Add** to open the **Comparison Item Properties** dialog box.
10. Select **ACCEPTED AMOUNT** and an operator of + (**Addition**), and then click **OK**.
11. Click **Add** to reopen the **Comparison Item Properties** dialog box.
12. Select **REJECTED AMOUNT** and an operator of (**none**), and then click **OK**.

## Tutorial 1

### Value Validation

13. Verify that your final comparison definition looks like the following:

The screenshot shows the 'Comparison Properties' dialog box. The 'Name' field is 'INPUT-OUTPUT CHECK', 'Initial State' is 'Active-rule is processed initially', 'Description' is 'Compare beginning amount to current amount', 'Comparison Type' is 'Numeric', and 'Decimal Positions' is '0'. The 'Left Side of Comparison' contains 'BEGINNING AMOUNT'. The 'Right Side of Comparison' contains 'ACCEPTED AMOUNT + REJECTED AMOUNT'. The 'Formula Relation' section has 'Operator' set to 'Equal (EQ)', 'Tolerance Type' set to 'None', and 'Tolerance Value' set to '0'. Buttons for 'Add', 'Insert', 'Update', 'Remove', 'OK', 'Cancel', and 'Help' are visible at the bottom.

14. Click **OK** to return to the ACR/Instream Editor main window.

### Task 10: Define the Actions for the Comparison Rule

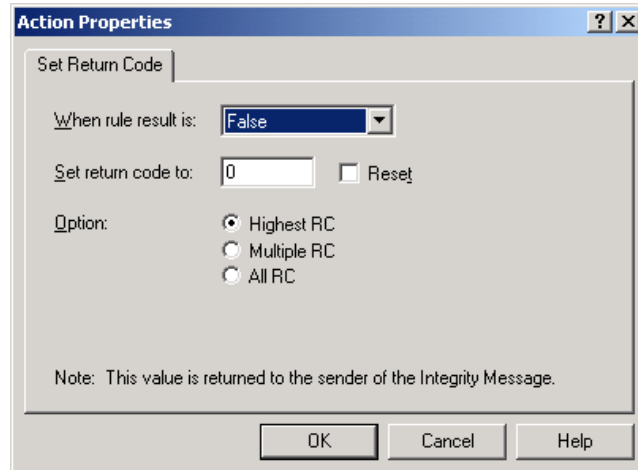
For each possible outcome of the comparison rule—TRUE or FALSE—you can define at least one action for ACR/Instream to take. In this sample application, you'll define one action. This action is for when the items are out of balance. An item is out of balance when the comparison you just defined is evaluated as FALSE. The action sends a return code to the application.

**Note:** An important concept of writing rules is purging an item from ACR/Instream's memory when it is no longer needed. You'll add an action in Tutorial 4 that will perform the purge. If this was the only comparison rule in your controls, you would need to add the action now to purge the item when its integrity is verified.

The ACR/Instream Editor main window should be showing before you continue.

1. Expand the Comparisons folder under RECORD REJECTED AMOUNT to display the INPUT-OUTPUT CHECK comparison.

2. Select the INPUT-OUTPUT CHECK comparison again and select **Edit** > **New** > **Set Return Code** to open the **Action Properties** dialog box for the Set Return Code action.

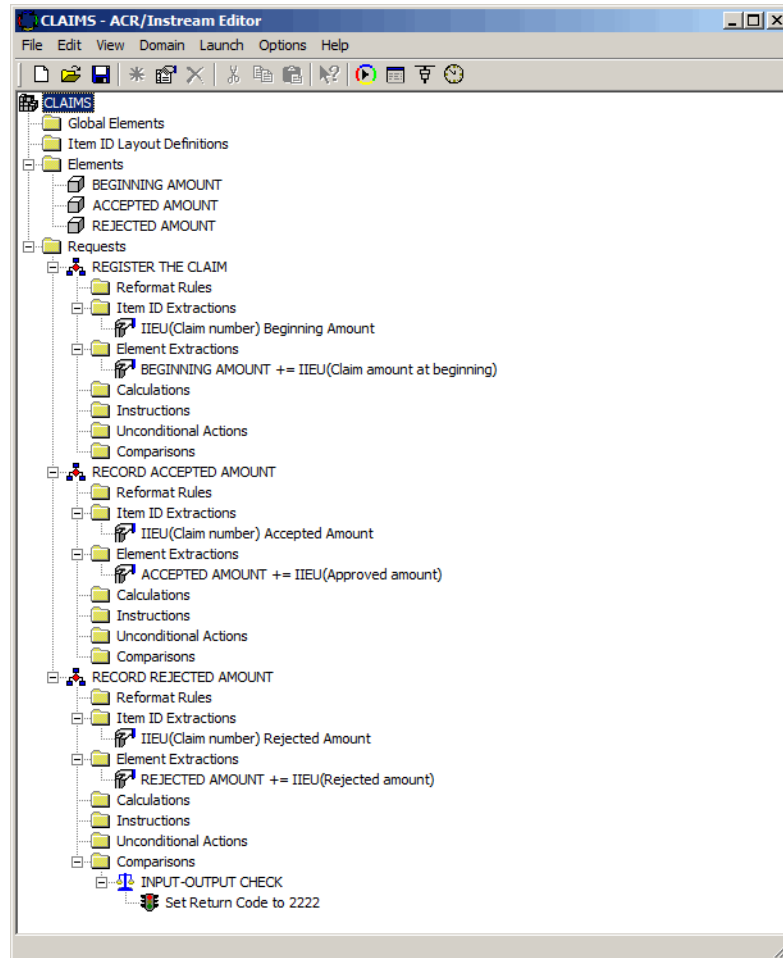


3. Enter **2222** for **Set return code to:** and click **OK**. Use the defaults of **FALSE** for **When rule result is:** and **Highest RC**.

## Tutorial 1

### Value Validation

4. Verify that your request definition looks like the following:



If you detect an error in your controls, it's easy to correct: double-click the item that needs correction. In the dialog box that opens, enter or select the correct information and click **OK**.

### Task 11: Save the Rules File

You're now ready to save your rules and go on to test them.

1. Select **File > Save As** and complete the **Save As** dialog box, naming your file **tutorial.igb**. You can use another name—be sure to write it down for later use in Tutorial 2.

Infogix recommends using the .igb extension for your file name and placing it in the following folder:

```
Program Files\Infogix\ACR Instream\Domain\tutorial.igb
```

You can use another location; however, placing it in the above folder simplifies the loading procedure in Tutorial 2. If you decide to change the location, you'll need to specify the new path in “Task 1: Load the Rules” on page 46.

If you do not have the above path on your Windows PC, search for the following file: run65l.bat. Substitute that path for the one above.

2. Do one of the following:
  - Exit the ACR/Instream Editor by selecting **File > Exit** or using any other Windows application exit method.
  - Go on to Tutorial 2 to learn how to load your new rules and test them using the ACR/Instream Player.

## Summary of the CLAIMS Item Group

This section describes the contents of the CLAIMS item group for this tutorial.

---

### Item Group: CLAIMS



**Elements:** BEGINNING AMOUNT (Extracted)  
ACCEPTED AMOUNT (Extracted)  
REJECTED AMOUNT (Extracted)



**Request Definition: REGISTER THE CLAIM**  
Purpose: Extracts data for AMOUNT AT BEGINNING



**Request Definition: RECORD ACCEPTED AMOUNT**  
Purpose: Extracts data for ACCEPTED AMOUNT



**Request Definition: RECORD REJECTED AMOUNT**  
Purpose: Extracts data for REJECTED AMOUNT  
Performs the INPUT-OUTPUT CHECK  
Comparison: INPUT-OUTPUT CHECK  
Formula: BEGINNING AMOUNT EQ ACCEPTED AMOUNT + REJECTED AMOUNT  
False action: Set Return Code to 2222

## Tutorial 1

---

### *Summary of the CLAIMS Item Group*

# Tutorial 2

This tutorial introduces the concepts and procedures for loading and testing the rules file you created in Tutorial 1. If you have other rules files created with the ACR/Instream Editor, you can use the procedures in this tutorial to load and test the rules—substitute your item group file name for the one in the procedures.

After you complete this tutorial, you'll be able to:

- Load your rules file into the ACR/Instream rules definition file
- Start and stop the ACR/Instream domain on Windows
- Read and interpret the Rules Definition File Update report
- Test the rules using the ACR/Instream Player

## Loading and Testing Rules

Loading the rules involves running a program that takes your rules file and puts it into the rules definition file. Testing is performed with the ACR/Instream Player. The ACR/Instream Player is a tool for developing your rules you can use to assess how well your rules work. In this tutorial, the proper functioning of the rules includes generating an Integrity Check report when the rules determine an integrity problem.

### Concepts for this Tutorial

This section introduces concepts you'll apply in this tutorial:

- Starting and stopping an ACR/Instream domain
- Rules definition file
- ACR/Instream Player, a testing tool
- Rules Definition File Update report
- Local mode and remote mode
- Connecting to ACR/Instream through middleware

#### **Starting and Stopping an ACR/Instream Domain**

ACR/Instream requires a startup procedure before it can process any integrity message. ACR/Instream does not require a startup for you to write or load rules. You do, however, need to start the ACR/Instream domain to run the ACR/Instream Player.

Starting and stopping ACR/Instream might be the responsibility of the ACR/Instream administrator at your site. If the ACR/Instream domain is installed on your Windows PC, you can use the **Start ACR/Instream Domain** and **Stop ACR/Instream Domain** icons, as described in this tutorial.

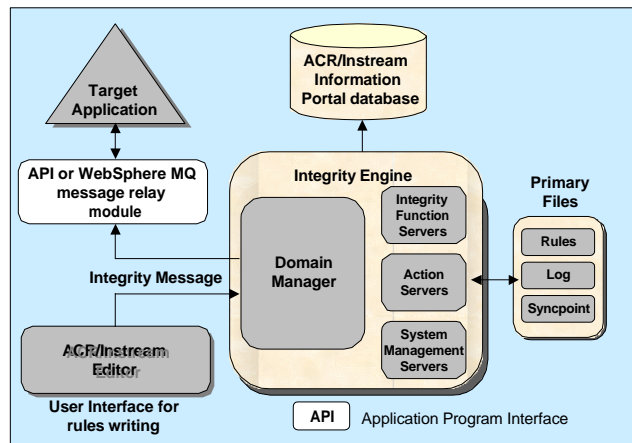
#### **Warm Starts and Cold Starts for the ACR/Instream Domain**

When the ACR/Instream domain is started, it uses a *warm start* or *cold start* parameter. A warm start reloads information about what happened in the past from a special file called a syncpoint file. This file contains an image of what was in ACR/Instream's memory at the time of the previous shutdown. The purpose of the warm start is to return to ACR/Instream's memory all data needed to continue processing as if it had never been shutdown.

The cold start, however, allows you to work with test files and develop rules iteratively. For example, at the end of this tutorial, you'll have certain information in memory. If you want to reload and retest (as described in ["Reloading and Retesting the Rules" on page 54](#)), you'll need the cold start. The tutorial teaches you to modify the startup procedure to perform a cold start.

## Rules Definition File

When your rules are ready to be tested, you load them into the rules definition file through the domain manager. For a real application, the APIs send integrity messages to the domain manager, which relies on the rules definition file.



## Testing Rules Using the ACR/Instream Player

The ACR/Instream Player reads an ACR/Instream log file of test integrity messages and sends the integrity messages to an ACR/Instream domain. In this manner, the ACR/Instream Player can substitute for your application while using real messages to test your rules.

The output values, return codes, and other data are displayed as the messages are executed. From the ACR/Instream Player view, you can assess the proper functioning of rules. For this tutorial, sample messages are provided that you can use with the ACR/Instream Player to test the rules you created in Tutorial 1.

## Rules Definition File Update Report

When you load your rules, ACR/Instream creates the Rules Definition File Update report. This report shows the conversion of your rules file into records that can be used by the rules definition file as well as the interpretation of those records.

The purpose of reviewing the report is to allow you to verify the rules were successfully loaded. The ACR/Instream Editor detects most errors as you are creating the rules; however, some types of errors cannot be detected until they are processed internally by ACR/Instream.

The tutorial shows you how to access the report and look for error messages.

## Tutorial 2

---

### *Loading and Testing Rules*

#### **Connecting through Middleware**

You can connect to an ACR/Instream domain on any platform from your Windows PC if you have the appropriate middleware, such as TCP/IP and WebSphere MQ. Your Windows PC may have more than one type of middleware, so you can select which to use. For example, you can select TCP/IP to connect to the ACR/Instream domain on Windows and WebSphere MQ to connect to the ACR/Instream domain on MVS.

To access the Connection Editor, double-click ACR/Connection Editor in your ACR/Instream Application folder to open the **Connection Editor** dialog box. Click the connection you want and click **Close**.

These tutorials show you how to connect to the ACR/Instream domain on your Windows PC. This type of connection does not require middleware.

### **Tasks for Tutorial 2**

In this tutorial, you will perform the following tasks:

1. Load the rules.
2. Review the Rules Definition File Update report.
3. Modify the startup file to perform a cold start.
4. Load the message file into the ACR/Instream Player.
5. Connect to the ACR/Instream domain.
6. Set the speed and start the playback.
7. Shutdown the ACR/Instream domain.

Keep the following in mind:

- Complete the tasks in order
- You can stop at any time; be sure to save your work
- Allow one-half hour to one hour to complete the tutorial

#### **Task 1: Load the Rules**

In tutorial 1, you used ACR/Instream Editor to create the rules. Now you need to move those rules into the rules definition file so you can test them.

These instructions are for an ACR/Instream domain on Windows only.

1. Access an MS-DOS command prompt window. To do this, click **Start**, point to **Programs**, and then click **Command Prompt**.
2. Navigate to your ACR/Instream program folder. The default path is the following:

```
C:\Program Files\Infogix\ACR Instream\domain
```

If you used the recommended location in “Task 11: Save the Rules File” on page 40, this folder now contains the loading command (run65l.bat) and your new rules file (tutorial.igb).

3. Enter the following command at the MS-DOS prompt in the domain folder:

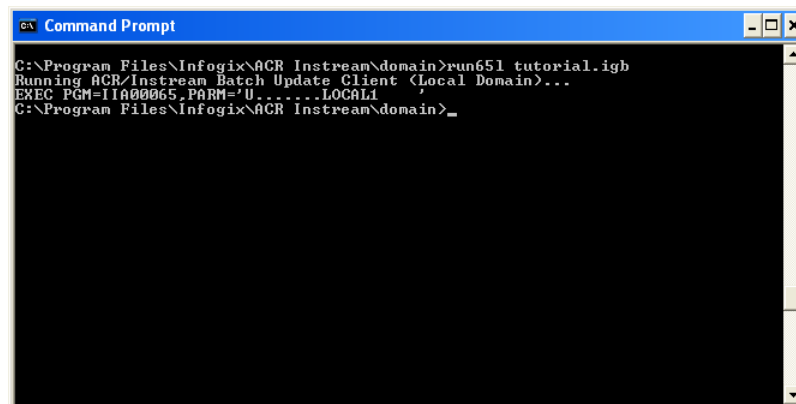
```
run65l tutorial.igb
```

The last character of run65l is a lowercase L. If you did not save your rules file (tutorial.igb) in the same directory as the run65l command, you will need to specify the path, as shown below:

```
run65l [drive:][path] tutorial.igb
```

The run65l command is a *local mode* command. Local mode means the ACR/Instream domain is not started. If your ACR/Instream domain is running, you would use the run65 command.

When you execute the run65l command, you’ll see an update as shown in the graphic below:



```
Command Prompt
C:\Program Files\Infogix\ACR Instream\domain>run65l tutorial.igb
Running ACR/Instream Batch Update Client (Local Domain)...
EXEC PGM=IIA00065,PARM='U.....LOCAL1
C:\Program Files\Infogix\ACR Instream\domain>
```

When run65l is executed, it generates the Rules Definition File Update report, which you review in the next task.

### Task 2: Review the Rules Definition File Update Report

In this task, you verify your new item group file has no errors by reviewing the update report. The ACR/Instream Editor checks most entries for internal consistency and valid values. However, certain types of errors can only be detected within a wider context than a single item group file. For example, if your item group name is already in use, ACR/Instream can only detect it during the rules definition file update.

## Tutorial 2

### *Loading and Testing Rules*

---

Therefore, you should always check this update report to verify the update completed successfully.

1. Open the file **IIAUPR.RPT** with any text editor, such as WordPad. The default location for this file is the following:

```
c:\Program Files (x86)\Infogix\ACR Instream\domain\IIAUPR.RPT
```

2. Review the report, which shows the output from the update. Each line will look similar to the following:

```
5 002BEGINNING AMOUNT                0000000                000
```

The first number (4 in the example) is the line number. The second number (002 in the example) is the record type. Each definition in your rules has a corresponding record type. In the above example, the record number (002) indicates an Item Group Element Record for an element named BEGINNING AMOUNT . The remaining data in the line is the definition of the element.

3. Scroll to the bottom of the report and look for the following phrase:

```
THE ITEM GROUP PASSED THE INTERNAL CHECKS WITH NO ERRORS -  
EXAMINE THE FOLLOWING RULES DEFINITION FILE INTEGRITY REPORT -  
RESOLVE ERRORS AS NECESSARY
```

Above this phrase is the Rules Definition File Update Report, which reports on items within the item group.

Below the phrase is the Rules Definition File Integrity Report, which reports on errors between item groups, data dictionaries etc. Every time you load a rules file, ACR/Instream checks for internal conflicts and reports on the current status. More information about these reports are in your ACR/Instream online Help.

If ACR/Instream detects errors, you'll see the following phrase:

```
UPDATE FAILED DUE TO                X ERRORS
```

You might receive this message if the ACR/Instream domain was started on your Windows desktop while you ran the local mode command. If this is the case, click the **Stop ACR Instream Domain** icon and return to the previous task to rerun the run65] command.

This tutorial assumes you have no errors in your item group file. However, if you do have an error now or in the future, you'll receive an error message along with an error message number. A list of error messages with descriptions is in your ACR/Instream online Help.

### Task 3: Modify the Startup File to Perform a Cold Start

Before you start the next task, you will need to modify your runim.bat to perform a cold start. The reason for this task is to allow the data for each tutorial to be ignored for subsequent tasks.

1. Open the file named runim.bat with any text editor, such as WordPad. The default location for this file is the following:

```
c:\Program Files\Infogix\ACR Instream\Domain\runim.bat
```

2. Locate the following line with the WARMDEBUG parameter:

```
SET PARMDAT=WARMDEBUG
```

3. Change the parameter to COLDDDEBUG:

```
SET PARMDAT=COLDDDEBUG
```

4. Save the file. You must use the runim.bat name.

---

**Note:** The normal startup of the ACR/Instream domain is a warm start. When you are finished with the tutorials, you must edit the startup file to perform a warm start. See [“The test file contains simulated problems. For this tutorial, the following problems should be detected:”](#)

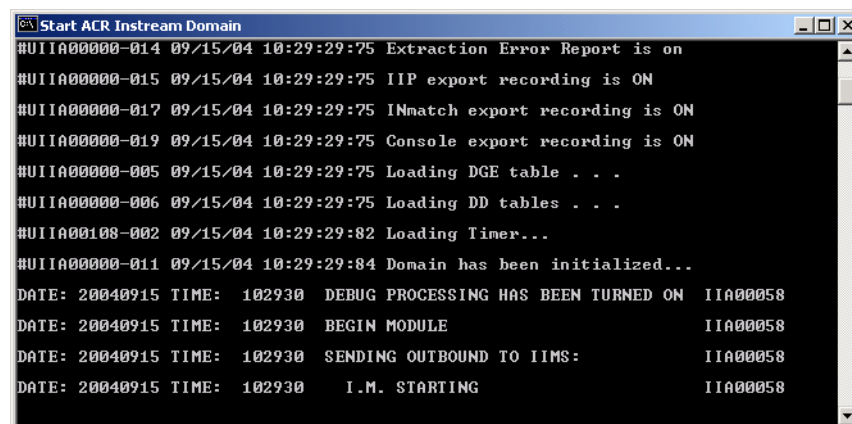
---

### Task 4: Load the Message File into the ACR/Instream Player

If you successfully updated the ACR/Instream domain with your rules, you're ready to learn how to use ACR/Instream Player to test your rules with the sample integrity messages.

1. Open the ACR/Instream application folder and double-click the **Start ACR Instream Domain** icon. Or, click **Start**, then select **Programs > Infogix > ACR Instream > Start ACR Instream Domain**.

An MS-DOS window opens and displays messages as the ACR/Instream domain starts.

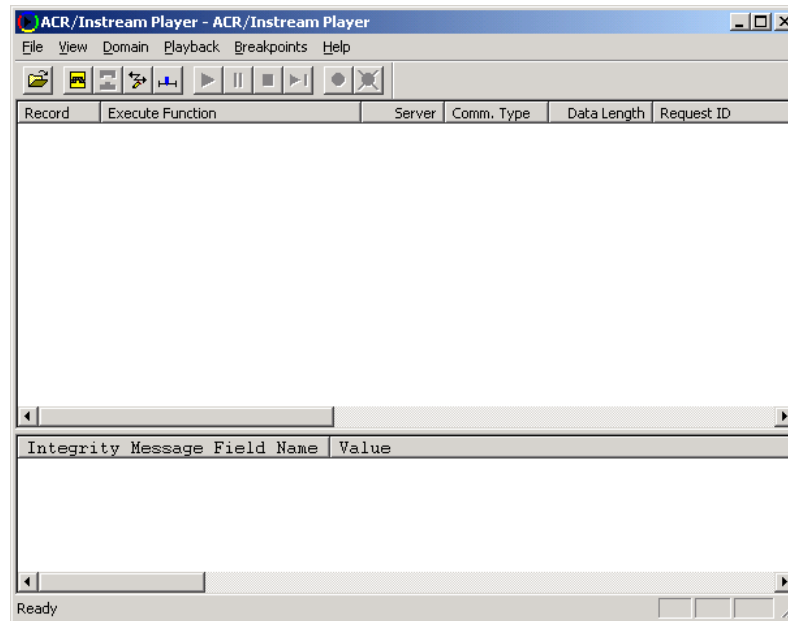


```
Start ACR Instream Domain
#UIIA00000-014 09/15/04 10:29:29:75 Extraction Error Report is on
#UIIA00000-015 09/15/04 10:29:29:75 IIP export recording is ON
#UIIA00000-017 09/15/04 10:29:29:75 INmatch export recording is ON
#UIIA00000-019 09/15/04 10:29:29:75 Console export recording is ON
#UIIA00000-005 09/15/04 10:29:29:75 Loading DGE table . . .
#UIIA00000-006 09/15/04 10:29:29:75 Loading DD tables . . .
#UIIA00108-002 09/15/04 10:29:29:82 Loading Timer...
#UIIA00000-011 09/15/04 10:29:29:84 Domain has been initialized...
DATE: 20040915 TIME: 102930 DEBUG PROCESSING HAS BEEN TURNED ON IIA00058
DATE: 20040915 TIME: 102930 BEGIN MODULE IIA00058
DATE: 20040915 TIME: 102930 SENDING OUTBOUND TO IIMS: IIA00058
DATE: 20040915 TIME: 102930 I.M. STARTING IIA00058
```

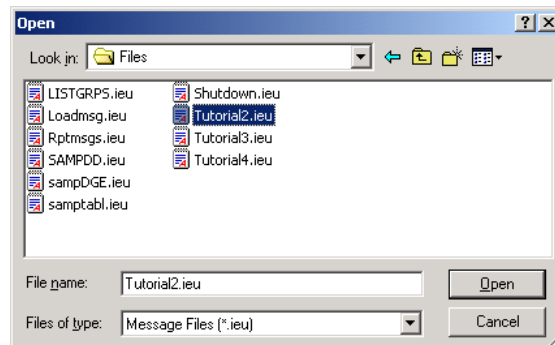
## Tutorial 2

### Loading and Testing Rules

2. Minimize the **Start ACR/Instream Domain** window.
3. Open the ACR/Instream application folder and double-click **ACR Instream Player**. Or, click **Start**, then select **Programs > Infogix > ACR Instream > ACR Instream Player**.



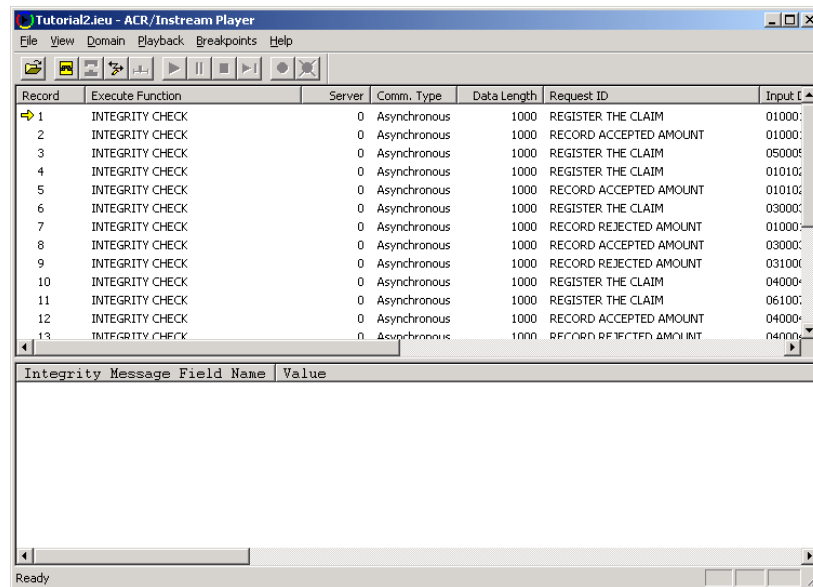
4. Select **File > Open** to open the **Open** dialog box.
5. Change the **Look in:** folder to **Files**, which is the default location for the tutorial files shipped on the installation media.




6. Select **tutorial2.ieu** from file list.

When you develop your rules, you will use your own ACR/Instream message file to simulate the integrity messages to be received from the API.

- Click **Open**. Your window should now look like the following.



The upper pane shows the contents of your message file. The record pointer  indicates the next record to be executed. Later, during playback, you'll see the record pointer move down the messages.

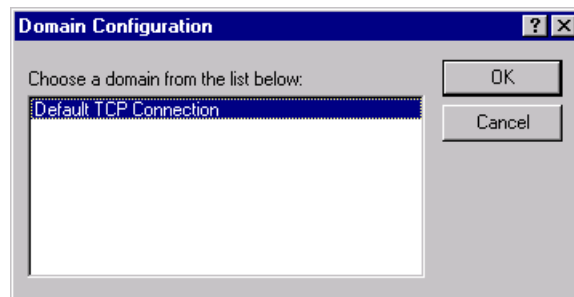
The message file pane is divided into columns that describe the integrity message. If you are using a message file that simulates the incoming integrity messages from an API, most messages will show only the INTEGRITY CHECK function in the Execute Function column.

The Input Data column contains the variable data that describes the data elements from the application. You might need to scroll to see all the data.

#### Task 5: Connect to the ACR/Instream Domain

You may not need to perform all these steps in the future. These steps will help you determine the easiest way for you to connect to the ACR/Instream domain.

1. Select **Domain > Configure...** to open the **Domain Configuration** dialog box. If you have access to more than one ACR/Instream domain, you can choose one from this dialog box.



If you have more than one selection, and you are uncertain which connection represents the ACR/Instream domain on your Windows PC, ask your ACR/Instream system administrator before continuing.

2. Select the domain in the **Domain Configuration** dialog box and click **OK**. If you have only one domain, it is the default selection.
3. Select **Domain > Connect**. The **ACR/Instream Player** connects to your ACR/Instream domain. If you have only one domain from which to select, this is the only step you need to perform in the future.

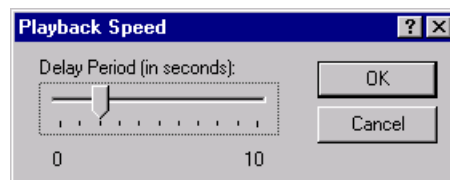


#### Task 6: Set the Speed and Start the Playback

Now that you've connected to the ACR/Instream domain, you're ready to use the ACR/Instream Player.



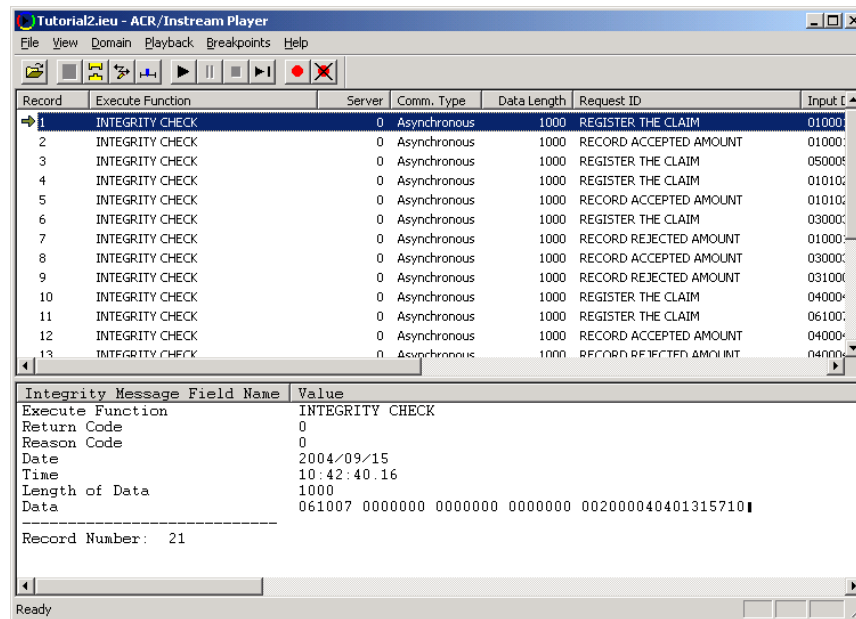
1. Select **Playback > Speed** to display the **Playback Speed** slider.



2. Move the slider to the right to increase the delay to the number of seconds of your choice and click **OK**.

When you first start testing, you might want a large delay period so you can monitor the processing. Later, with large message files to work with, you can decrease the delay period to speed up processing.

- ▶ 3. Select **Playback > Go**.



■ Watch the output in the lower pane. To stop the playback, select **Playback > Stop**. Or, you can select **Playback > Pause**.

As an alternative, you can use **F10** to execute a single message at a time.

4. Review the output values in the lower output pane. Two of the claims in the tutorial generate a return code.

The integrity message file provided with the tutorials includes messages that simulate problems. For this tutorial, the following problems should be detected:

Record number 9	Claim Number 031000	Return code of 2222	Fails balancing test: missing both messages for register the claim and accepted amount
Record number 16	Claim Number 050005	Return code of 2222	Fails balancing test: the amount at the beginning does not equal the accepted amount plus the rejected amount.

If you receive a reason code of 3037, the middleware connection to the ACR/Instream domain is not in place for the loading to complete. First verify that you clicked the Start ACR/Instream Domain icon.

If you need to change or correct your tutorial.igb file, be sure to read "[Reloading and Retesting the Rules](#)" below.

## Tutorial 2

### *Loading and Testing Rules*

---

#### **Task 7: Stop the ACR/Instream Domain**

By stopping the ACR/Instream domain, you'll ensure a cold start for the next tutorial, thus deleting items about this tutorial from ACR/Instream's memory.

1. Stop the ACR/Instream domain by double-clicking the **Stop ACR Instream Domain** icon in the Application folder. Or, click **Start**, then select **Programs > Infogix > ACR Instream > Stop ACR Instream Domain**.
2. Exit ACR/Instream Player by selecting **File > Exit** or using any other Windows application exit method.

Go on to Tutorial 3 to learn how to modify your rules to perform transaction lifecycle monitoring.

### **Reloading and Retesting the Rules**

If you want to retest the rules, you need to perform the following steps first:

1. Stop the ACR/Instream domain if necessary: double-click **Stop ACR Instream Domain** in the Application folder or click **Start**, then select **Programs > Infogix > ACR Instream > Stop ACR Instream Domain**.
2. Start the ACR/Instream domain: double-click **Start ACR Instream Domain** in the Application folder or click **Start**, then select **Programs > Infogix > ACR Instream > Start ACR Instream Domain**.

When you restart the ACR/Instream domain with the cold start parameter, it does not restore items into ACR/Instream's memory from the previous execution of ACR/Instream.

After you perform the steps above, you can reload your rules as described in "Task 1: Load the Rules" on page 46 or retest using the ACR/Instream Player. You do not need to reload your rules unless you returned to the ACR/Instream Editor to change or correct your tutorial.igb file.

# Tutorial 3

This tutorial continues the sample application you started in tutorial 1 by building your processing strategy to include conditional processing. While completing it, you'll create an application that monitors the processing time between procedures.

After you complete this tutorial, you'll be able to:

- Define rules that are activated only when certain conditions are met
- Define instructions for the application administrator for conditions that require attention
- Print the Integrity Check report

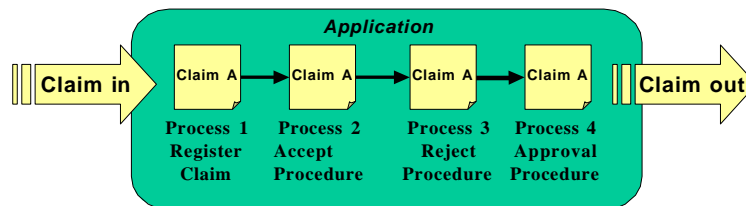
## Transaction Lifecycle Tracking

ACR/Instream can determine transaction time from process to process and verify that the system is operating properly. It ensures that processing takes place in a particular time frame and identifies if, and where, transactions are being held up.

This tutorial shows you how to write rules that determine the elapsed time between processing points and compare the elapsed time to a predefined amount of time.

### Sample Application

In this extension of tutorial 1, the control need is to monitor the time between the accept procedure and the reject procedure to make sure it is not excessive. In this application, the elapsed time between Process 2 and Process 3 should not exceed a maximum of 2-1/2 hours. By detecting excessive processing time, you can identify areas for improvement and perhaps identify problems that affect your customers.



The processing time between Process 2 and Process 3 must be less than a predefined amount of time

## Tutorial 3

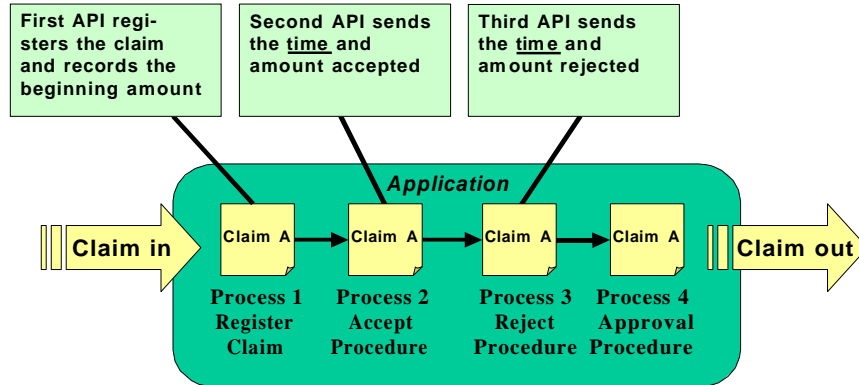
### Transaction Lifecycle Tracking

The control solution is to calculate the processing time and compare it to a predefined amount. When the comparison rules detect a claim that has taken an excessive amount of time, ACR/Instream performs the following actions:

- Generate an Integrity Check report
- Write a prewritten instruction for the application administrator

#### APIs for this Tutorial

For this tutorial, you can assume the same APIs are inserted into the application as you used in Tutorial 1. These APIs have the same message layouts as the ones you used in Tutorial 1. For this tutorial, you will define how to extract the time elements from the integrity message so you can calculate the processing time.



#### Concepts for this Tutorial

This section introduces concepts you'll apply in this tutorial:

- Conditional processing (active and inactive rules)
- Integrity Check reports
- Write Instruction action

#### Conditional Processing (Active and Inactive Rules)

When you develop the comparison rules for your application, you will find that sometimes a rule is needed only under certain conditions. For effective and efficient rules performance, you need to write rules that are processed only under those conditions.

To achieve the conditional processing, you can define the comparison rules as active or inactive. Active rules are always evaluated. Inactive comparisons can be activated by the Activate Rule action. As with other actions, the Activate Rule action can be processed on a TRUE or FALSE evaluation of a rule.

Active and inactive comparisons provide an easy if-then type of evaluation. For example, if the evaluation says the process is NOT OK, then perform the evaluation that detects certain problems.

#### **Integrity Check reports**

Integrity Check reports show the state of a controlled object and the results of the application of a request definition to that object. The reports can be printed as an action for any comparison rule. Typically, you would request a report to be printed when the rule evaluation indicates a problem.

#### **Write Instruction Action**

The Write Instruction action is a way for you to communicate with an application administrator. This action causes instructions, written in advance and stored with the request definition, to be printed on the Integrity Check report.

Note that if you do not also specify the Produce Report action, the Write Instruction has no effect.

### **Processing Strategy**

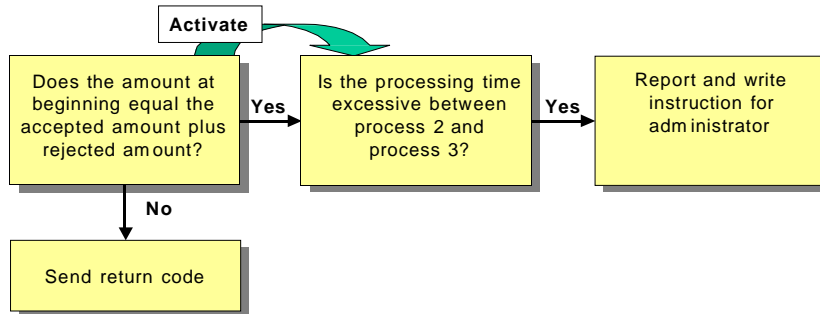
In the first tutorial, you checked for an out-of-balance condition for the claim when it completed Process 3, reject procedure. In this tutorial, you add an additional comparison that checks the amount of time between Process 2 and Process 3.

## Tutorial 3

### Transaction Lifecycle Tracking

---

But it's not necessary to test for excessive processing time unless the first comparison determines that the claim is in balance. To achieve this processing strategy, you can define a comparison rule to become active only when the claim is in balance.



***The check for excessive processing time is only performed if the claim passes the balancing test.***

The Activate Rule action is part of the first comparison rule that determines if the postings are in balance. That comparison rule uses the following formula that you created in Tutorial 1:

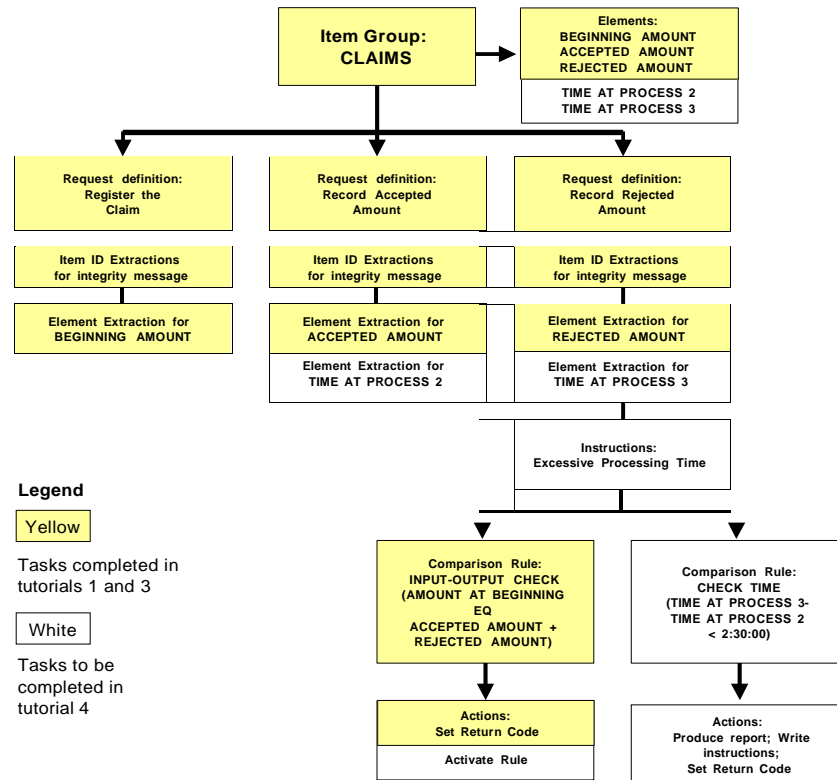
```
BEGINNING AMOUNT EQ (ACCEPTED AMOUNT + REJECTED AMOUNT)
```

If this comparison is evaluated as TRUE, the second comparison rule is activated that tests for the processing order. In this case, the following formula is applied that you will create in this tutorial:

```
(TIME AT PROCESS 3 - TIME AT PROCESS 2) < 2:30:00
```

If this formula is evaluated as FALSE, meaning the processing time exceeds the maximum desired of 2-1/2 hours, a report is generated for the application administrator.

For this tutorial, your item group structure will look like the following when your tutorial is complete:



## Tasks for Tutorial 3

You will perform the following tasks:

1. Open the CLAIMS item group.
2. Add new elements to the CLAIMS item group.
3. Define the extraction for the new elements.
4. Define the instructions for the application administrator.
5. Add the inactive CHECK TIME comparison rule.
6. Add the actions for the comparison rule.
7. Define the Activate Rule action.
8. Save, load, and test the rules.
9. Review your Integrity Check report.

## Tutorial 3

### *Transaction Lifecycle Tracking*

---

Keep the following in mind:

- Complete the tasks in order.
- The tasks assume the knowledge you gained in Tutorial 1 and 2.
- The tasks build on the sample application you created in Tutorial 1.
- Allow one-half hour to one hour to complete the tutorial.

#### **Task 1: Open the CLAIMS Item Group**

In this task, you return to the CLAIMS item group you created in Tutorial 1. The CLAIMS item group is saved in the file named tutorial.igb

1. Access ACR/Instream Editor as you learned to do in Task 1 of Tutorial 1.
2. Select **File > X tutorial.igb**, where X is a selection number in your **File** menu command list. For example, if the tutorial.igb is the last file you accessed with ACR/Instream Editor, then select **File > 1 tutorial.igb**.

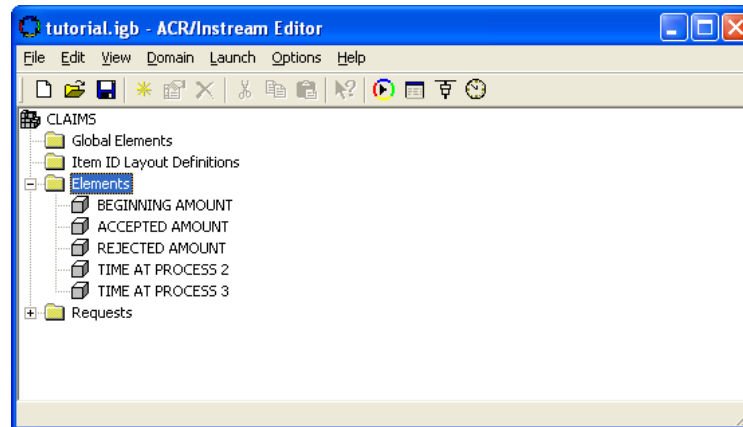
#### **Task 2: Add New Elements to the CLAIMS Item Group**

As in the first tutorial, you need to define elements to extract data from the integrity messages. These new elements extract the time for Process 2 and Process 3.

1. Select the Elements folder then select **Edit > New** to open the **Element Properties** dialog box.
2. Complete the **Element** dialog box for the first element and click **OK**.  
Name: TIME AT PROCESS 2  
Type: Date/Time  
Display: Date

Notice that this dialog box changes when you change the Type from numeric to Date/Time. You can no longer specify decimals—instead you can select a display type. If you were working with days, you could specify a display type of days.

- Repeat the above step for the second element, naming it TIME AT PROCESS 3. Use the type of Date/Time and a Display of Date. Your ACR/Instream Editor main window should look like the following:



### Task 3: Define the Extraction for the New Elements

You use the same data dictionary as you did in Tutorial 1 to define the element extraction.

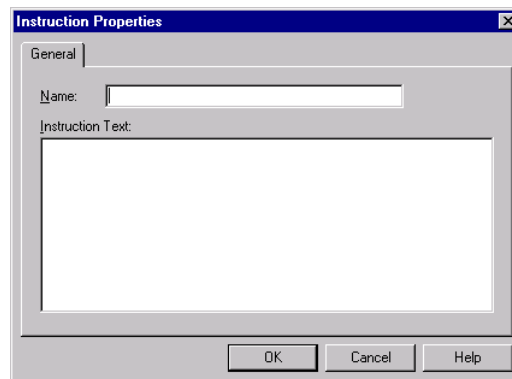
- Expand the Requests section.
- Expand the RECORD ACCEPTED AMOUNT section to display the request definition components.
- Select the Element Extractions folder and then select **Edit > New**.
- Enter or select the following data in the **Element Extraction Properties** dialog box:  
Element Name: TIME AT PROCESS 2  
Dictionary item: Current time
- Expand the RECORD REJECTED AMOUNT section to display the request definition components.
- Select the Element Extractions folder and then select **Edit > New**.
- Enter or select the following data in the **Element Extraction Properties** dialog box:  
Element Name: TIME AT PROCESS 3  
Dictionary item: Current time

You can use the same data dictionary item for both elements because in this sample application, the integrity messages from all APIs are structured exactly the same. That is, the location of the time stamp in each message is in the same location.

#### Task 4: Define the Instructions for the Application Administrator

For this application, instructions need to be printed on a report for the application administrator that describes the problem. By defining an instruction at the request definition level, you can use it in any comparison rule associated with that request definition. In other words, you could use it repeatedly for similar situations.

1. Expand the RECORD REJECTED AMOUNT section to display the request definition components, if necessary.
2. Select the Instructions folder and then select **Edit > New** to open the **Instruction Properties** dialog box.



In this dialog box you can define text that will print on the Integrity Check report. In a later task, you will assign these instructions as an action.

3. Enter **Excessive Processing Time** as the name.
4. Enter **Investigate excessive processing time for this claim** as the instruction text and click **OK**.

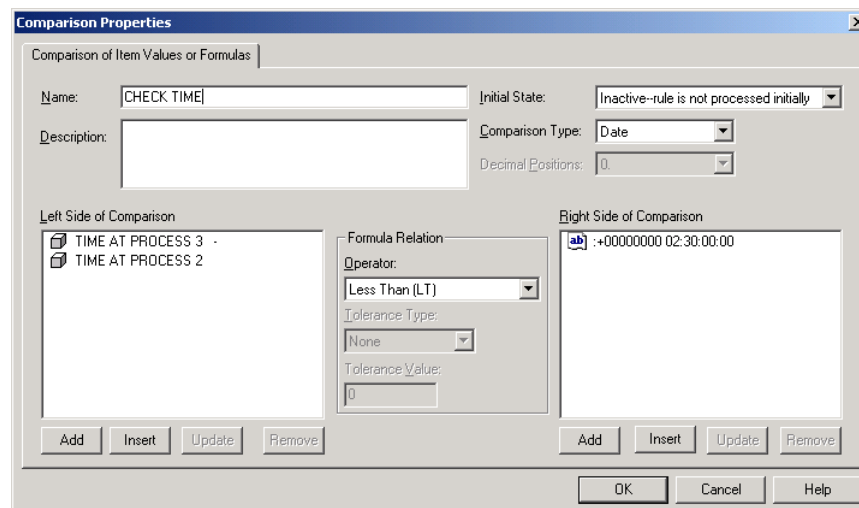
This text will print on the Integrity Check report.

#### Task 5: Add the Inactive CHECK TIME Comparison Rule

In this task, you add the CHECK TIME comparison rule to an existing request definition. In a later task, you can reference this new comparison rule.

1. Expand the RECORD REJECTED AMOUNT section to display the request definition components.
2. Select the Comparisons folder, and then select **Edit > New**.
3. Select **Comparison of item values or formulas** as your **Rule Type**. Click **OK**.

4. Enter the following data in the **Comparison Properties** dialog box:  
 Name: CHECK TIME  
 Initial State: Inactive--rule is not processed initially  
 Comparison Type: Date  
 Operator: Less Than (LT)  
 By specifying an initial state of inactive, ACR/Instream will not evaluate this comparison unless it is activated by the Activate Rule action.
5. Click in the window labeled **Left side of comparison**.
6. Click **Add** and select **TIME AT PROCESS 3**. Select an operator of **(Subtraction)**, and click **OK**.
7. Click **Add** and select **TIME AT PROCESS 2**. Select an operator of **(none)**, and click **OK**.
8. Click in the window labeled **Right side of comparison**.
9. Click **Add** and select a **Type** of **Days**. Note the tab changes to show a text box in which you can enter the literal.
10. Enter a literal value of **+00000000 02:30:00:00**, select an operator of **(none)**, and click **OK**.
11. Verify that your comparison definition looks like the following:



12. Click **OK** to return to the ACR/Instream Editor main window.

## Tutorial 3

### Transaction Lifecycle Tracking

---

#### Task 6: Add the Actions for the Comparison Rule

In this task, you define what happens when the CHECK TIME comparison rule is evaluated as TRUE or FALSE. In this sample application, a FALSE result indicates the processing time for the claim between Process 2 and Process 3 exceeded the maximum desired.

The ACR/Instream Editor main window must be showing before you continue.

1. Enter the inactive rule actions for the CHECK TIME comparison rule by selecting **Edit > New**, then selecting the appropriate actions.

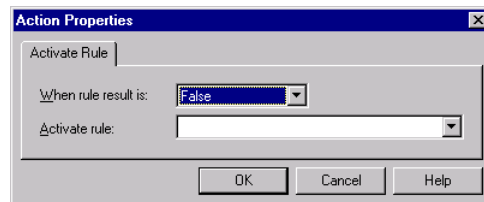
The table below provides the data you need to define these actions. For step-by-step instructions for defining actions, see “[Task 10: Define the Actions for the Comparison Rule](#)” on page 38.

Action type	When rule result is	Additional data
Write Instruction	FALSE	Click <b>Excessive Processing Time</b> in the Write Instruction list
Produce Report	FALSE	
Set Return Code	FALSE	2525

#### Task 7: Define the Activate Rule Action

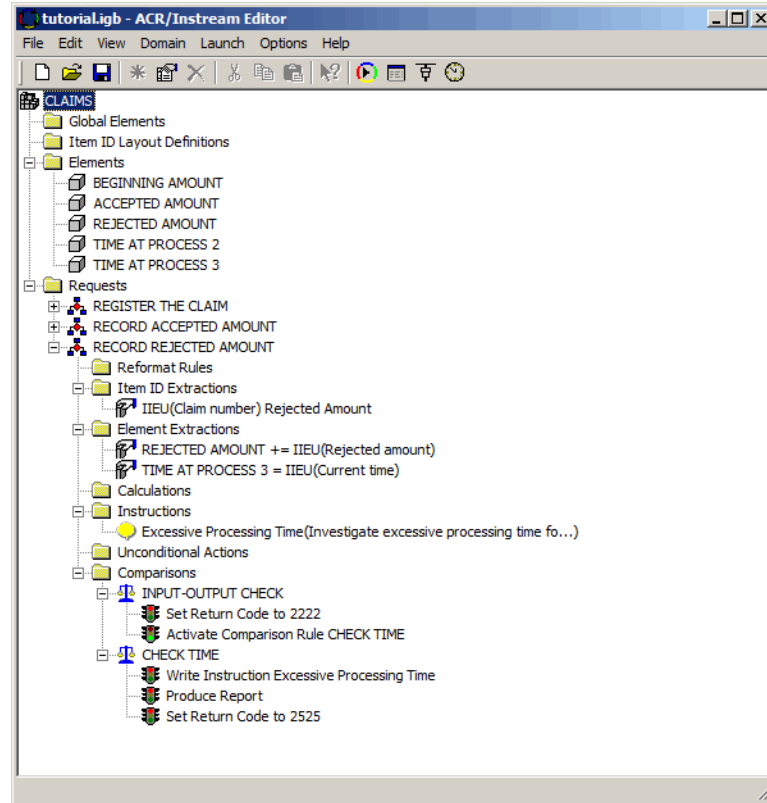
The ACR/Instream Editor main window must be showing before you continue.

1. Select the **INPUT-OUTPUT CHECK** comparison rule and then select **Edit > New > Activate Rule** to open the **Action Properties** dialog box.



2. Select **TRUE** for **When the result is:**.

3. Select **CHECK TIME** from the **Activate rule** list. Your rule should look like the following:



### Task 8: Save, Load, and Test the Rules

This task repeats what you learned in the previous tutorials, so you can test the new rules and generate an Integrity Check report.

1. Stop the ACR/Instream domain, if you did not stop it at the end of Tutorial 2.
2. Save your rules and load them into the rules definition file as you learned to do in “Task 1: Load the Rules” on page 46. Be sure to check the IIAUPR.RPT report for errors.
3. Start the ACR/Instream domain and test your rules using the ACR/Instream Player as you learned to do in Tutorial 2. Be sure to use the **tutorial3.ieu** message file.

Record number 14 will produce a return code of 2222 for an out-of-balance condition and record number 20 will produce a return code of 2525 and cause the Integrity Check report to be generated.

#### **Task 9: Review Your Integrity Check Report**

If you used the ACR/Instream Player to test your rules using the message file named tutorial3.ieu, an Integrity Check report was generated and placed on your PC. That's because the integrity message file was designed so that one claim, 091007, has excessive processing time between the two procedures.

1. Stop the ACR/Instream domain. The Integrity Check report is not available until the domain is shut down.
2. Open the Integrity Check report named **iiarpta.rpt** in any text editor, such as WordPad. The default location for this file is the following:

```
C:\Program Files\Infogix\ACR Instream\Domain\Files\iiarpta.rpt
```

An example of the Integrity Check report is shown on the next page.

3. Review the Integrity Check report.
  - a. Find the formula that calculated the time between process 3 and process 2.
  - b. Find the instructions printed at the bottom of the report.

Go on to Tutorial 4 to learn how to verify the process flow.

---

**Note:** If you are not continuing with the tutorials, you must edit the startup file to perform a warm start.

---

### **About the Integrity Check Report**

This report provides you the information you need to troubleshoot the discrepancies in the claim's state from one process to another:

- The specific request definition ID and item ID at the top of the report identify the specific business object with the discrepancy.
- Element values for the processing of the integrity message are shown in section 1. If an element value changed, the asterisk (\*) is printed at the end of the element name. In this case, REJECTED AMOUNT and TIME AT PROCESS 3 changed during the processing of the integrity message.
- The values used in the comparison formula are shown in section 5.
- Section 6 shows you how the comparison was evaluated and the actions that ACR/Instream took. In this example, ACR/Instream produced the report and printed the Excessive Processing Time instructions at the end.

## Tutorial 3

### Transaction Lifecycle Tracking

1DATE: 8/19/13 ACR/INSTREAM PAGE: 0001  
VERSION:5.5.0  
TIME: 16:09:48 INTEGRITY CHECK REPORT

REQUEST ID = RECORD REJECTED AMOUNT  
ITEM GROUP = CLAIMS  
ITEM ID = 091007

```
=====
1.2 ITEM ELEMENT VALUES                EXTRACTED                STORED
=====
BEGINNING AMOUNT                        0                        1,210
ACCEPTED AMOUNT                          0                        605
REJECTED AMOUNT                          * 605                    605
TIME AT PROCESS 2                        00/00/00                 04/04/00
                                         00:00:00:00             01:31:57:10
TIME AT PROCESS 3                          * 04/04/00                 04/04/00
                                         06:55:57:10             06:55:57:10
PROCESS COUNT                              0                        3
=====
```

```
=====
5. COMPARISONS                          LHS                      RHS
=====
INPUT-OUTPUT CHECK                       1,210                    1,210
FORMULA: BEGINNING AMOUNT EQ ACCEPTED AMOUNT + REJECTED AMOUNT

CHECK TIME                                00/00/00                 05:24:00:00
                                         00/00/00                 02:30:00:00
FORMULA: TIME AT PROCESS 3 - TIME AT PROCESS 2 LT 00000000 DAYS 02:30:00:00
=====
```

```
=====
6. ACTIONS TAKEN
=====
```

INPUT-OUTPUT CHECK IS TRUE  
EVALUATE COMPARISON=CHECK TIME

CHECK TIME IS FALSE  
INSTRUCTIONS=Excessive Processing Time  
PRODUCE REPORT  
SET HIGHEST RETURN CODE TO 2525

Excessive Processing Time  
Investigate excessive processing time for this claim

## Tutorial 3

### Transaction Lifecycle Tracking

---

## Summary of the CLAIMS Item Group for Tutorial 3

This section describes the contents of the completed item group for tutorial 3. Components added in this tutorial are indicated with an asterisk (\*).

---

### Item Group: CLAIMS



**Elements:**

- BEGINNING AMOUNT (Extracted)
- ACCEPTED AMOUNT (Extracted)
- REJECTED AMOUNT (Extracted)
- \*TIME AT PROCESS 2 (Extracted)
- \*TIME AT PROCESS 3 (Extracted)



#### **Request Definition: REGISTER THE CLAIM**

Purpose: Extracts data for BEGINNING AMOUNT



#### **Request Definition: RECORD ACCEPTED AMOUNT**

Purpose: Extracts data for ACCEPTED AMOUNT and \*TIME AT PROCESS 2



#### **Request Definition: RECORD REJECTED AMOUNT**

Purpose: Extracts data for REJECTED AMOUNT and \*TIME AT PROCESS 2

Performs the INPUT-OUTPUT CHECK

\*Checks for excessive time between processes

\*Instructions: "Investigate excessive processing time for claim"

Comparison: INPUT-OUTPUT CHECK

Formula: BEGINNING AMOUNT EQ ACCEPTED AMOUNT + REJECTED AMOUNT

\*True action: Activate comparison rule CHECK TIME

False action: Set Return Code to 2222

\*Comparison: CHECK TIME (inactive)

\*Formula: (TIME-AT-PROCESS-3 - TIME-AT-PROCESS-2) LT 2:30:00

\*False action: Write Instruction

\*False action: Produce Report

\*False action: Set Return Code to 2525

# Tutorial 4

This tutorial continues the sample application from the previous tutorials by advancing your processing strategy to validation process flow. You will use additional comparison rules strategies and different techniques for obtaining element values.

After you complete this tutorial, you'll be able to:

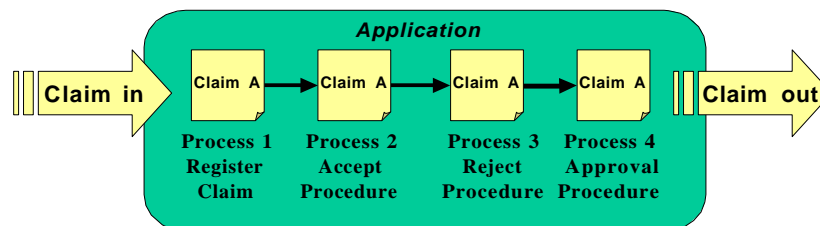
- Define a calculation of an element value
- Increment an element value to count the number of transactions

## Validating the Process Flow

In real-time systems, transactions have a defined path in which they must be processed. Occasionally a process can be skipped due to system failures or interruptions. Process flow validation controls assure that information is processed in a particular way. This tutorial shows you one method of validating the process flow.

### Sample Application

This sample application builds on the one you used in the previous tutorials. The control need is to verify that a claim completes all processes before it exits the application. The processes include the approval procedure, Process 4.



Each claim must complete the four processes before exiting the application.

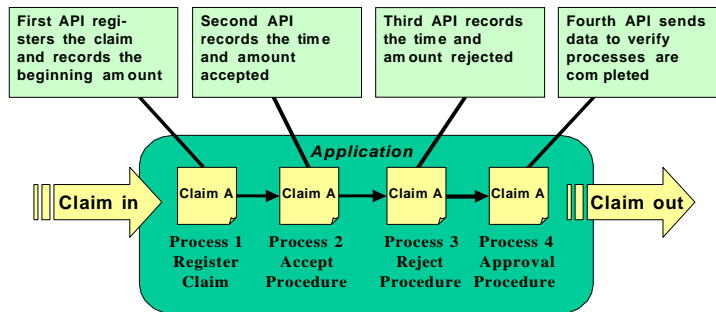
The control solution for this application is to count the number of processes each claim has completed. Before it exits the application, a comparison is made of the required count and the actual count. If the controls determine that the processes were not all complete, a return code is sent back to the application.

## Tutorial 4

### Validating the Process Flow

#### APIs for this Application

For this tutorial, you can assume a fourth API is added to the three you used in the previous tutorials. The fourth API sends data that indicates the approval procedure is complete and the claim is ready to exit the application.



#### Concepts for this Tutorial

This section introduces the following concepts you'll apply in this tutorial:

- Calculations
- Element storage and incrementing
- Purging item IDs

#### Calculations

You can define a calculation formula to add, subtract, multiply, and divide using either elements just extracted or elements stored from earlier processing.

For this application, the calculation formula is used to track the total number of processes that a claim has completed. It does this simply by adding a value to the element every time an API sends an integrity message indicating the process is complete.

#### Element Storage and Incrementing

To save the calculation result for use in a comparison, you'll specify an element in which to store the value. For this specific application, you'll store the result back to the element to achieve the *incrementing* action.

$$\text{PROCESS COUNT} + 1 = \text{New Value}$$

New value is stored in  
PROCESS COUNT

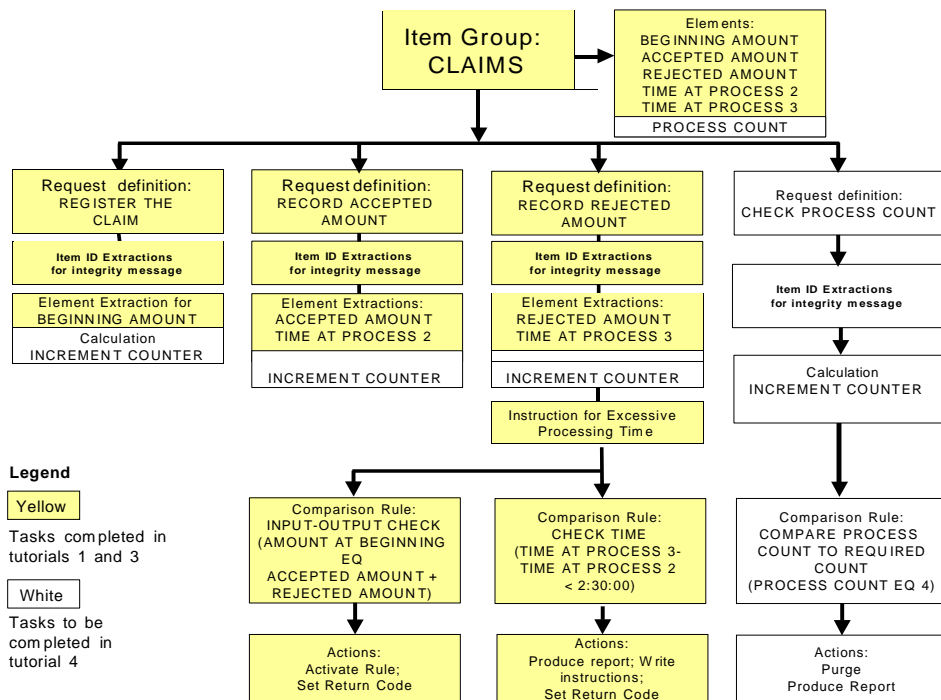
### Purging Item IDs

Data about specific business objects are accumulated in ACR/Instream’s memory for use with evaluations. Item IDs must be purged when they are no longer needed. Purging is an action that can be performed as the result of a comparison rule evaluation. A typical use of the Purge Item IDs action is when a rule is evaluated that indicates the target application is running properly.

### Processing Strategy

In this application, each claim has a counter that is incremented every time an API sends a message from a process. When the fourth API sends an integrity message indicating that Process 4 is complete, the counter must be equal to four if the claim completed all processes.

The diagram below shows how this new strategy combines with the strategy from the previous tutorial.



### Tasks for Tutorial 4

The sample integrity messages you used in the previous tutorials can also be used with this tutorial. You will perform the following tasks:

1. Add a new element to the CLAIMS item group.

## Tutorial 4

### *Validating the Process Flow*

---

2. Add a calculation to REGISTER THE CLAIM request definition.
3. Add the calculation to the other request definitions.
4. Add a new request definition to check the count.
5. Copy the calculation to count the number of processes.
6. Add a comparison rule to check the process count.
7. Add the actions for the comparison rule.
8. Load and test the rules.
9. Modify the startup file to perform a warm start.

Keep the following in mind:

- Complete the tasks in order.
- The tasks assume the knowledge you gained in the previous tutorials.
- The tasks build on the sample application you created in the previous tutorials.
- Allow one to two hours to complete the tutorial.
- For a list of the complete item group contents, see “[Summary of the CLAIMS Item Group for Tutorial 4](#)” on page 80.

#### **Task 1: Add a New Element to the CLAIMS Item Group**

This task returns to the item group you created in Tutorial 1 to add an element that can be used to calculate the number of completed processes for each claim.

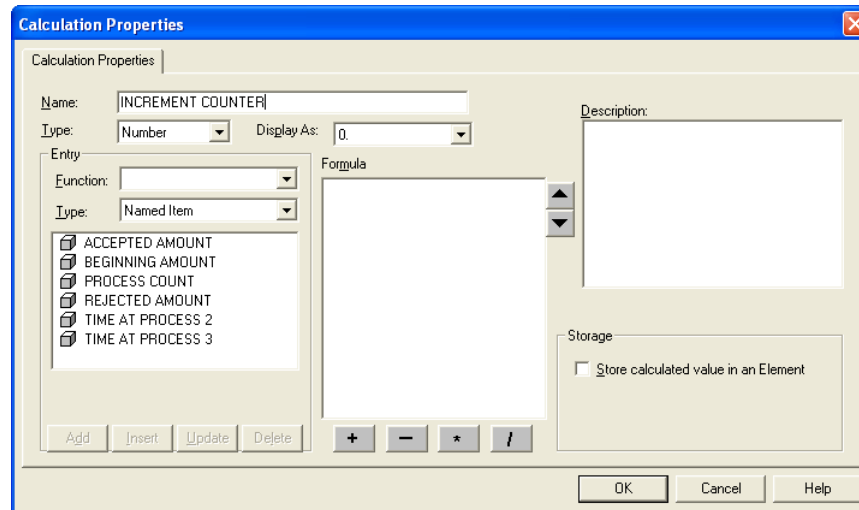
1. Access ACR/Instream Editor as you learned to do in Task 1 of Tutorial 1.
2. Select **File > X tutorial.igb**, where X is a selection number in your **File** menu command list. For example, if the tutorial.igb is the last file you accessed with ACR/Instream Editor, then select **File > 1 tutorial.igb**.
3. Add the new element **PROCESS COUNT** and click **OK**.  
Use the default type of numeric and the default decimals of 0.

This new element will be used to store the calculation you’ll define in the next task. Unlike the other elements you defined in previous tutorials, you will not need to define an element extraction for this new element.

## Task 2: Add a Calculation to REGISTER THE CLAIM Request Definition

In this task you define a calculation that increments a counter.

1. Expand the REGISTER THE CLAIM request definition section.
2. Select **Calculations > New** to open the **Calculation Properties** dialog box.

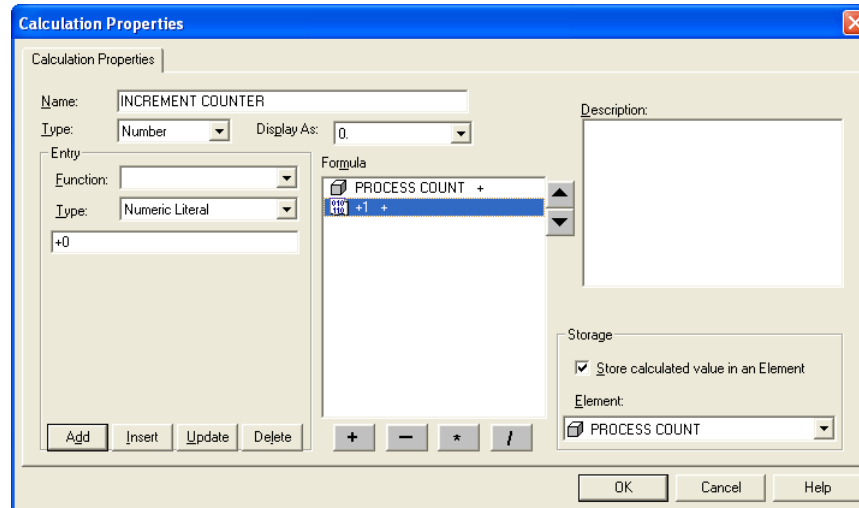


3. Enter **INCREMENT COUNTER** as the name. Use the defaults for **Type** and **Display AS**.
4. Enter a check mark for **Store calculated value in an Element**.
5. Select **PROCESS COUNT** from the **Element** list below the **Store Calculated value in an Element**.
6. Select **PROCESS COUNT** in the pane on the left and click **Add**. Then select an operator of + (addition). These element values are listed when the **Type** in the drop-down list above it is **Named Item**, which is the default.
7. Select a **Type** of **Numeric Literal**. A text box in which you can enter the literal is now available.

## Tutorial 4

### Validating the Process Flow

8. Enter a Literal Value of **+1** and click **Add**. Your formula should look like the following:



9. Click **OK** to return to the ACR/Instream Editor main window.

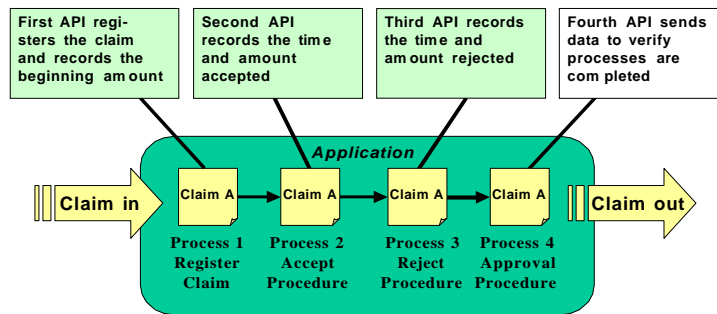
### Task 3: Add the Calculation to Other Request Definitions

When you complete this task, you will have a calculation element that is updated by all incoming integrity messages from three APIs for processes 1, 2, and 3.

1. Right-click the new calculation named INCREMENT COUNTER and then select **Copy**.
2. Expand the request definition RECORD ACCEPTED AMOUNT.
3. Right-click the Calculations and then select **Paste**. The calculation is copied to the new location.
4. Repeat the paste into the Calculations folder for RECORD REJECTED AMOUNT.

**Task 4: Add a New Request Definition to Check the Count**

In this task, you add a request definition for the API that sends data when Process 4 is complete. The only data to be extracted from the integrity message sent by the API is the claim number, so ACR/Instream knows which claim has completed its processes.



The ACR/Instream Editor main window should be showing before you continue.

1. Select the Requests folder, then select **Edit > New** to open the **Request Properties** dialog box.
2. Enter **CHECK PROCESS COUNT** in the **Name** box. Again, be sure your request name matches exactly so the tutorial will work correctly.
3. Select **Tutorial** from the **Data Dictionary** list.
4. Enter **Increment counter and check process count for the claim** as the description and click **OK**.
5. Expand the new request definition, select the Item ID Extractions folder, and then select **Edit > New** to open the **Item ID Extraction Properties** dialog box.
6. Enter **Claim number** as the Item ID description.
7. Click Data Dictionary, select **Claim number** from the drop-down list, and then click **OK**.

**Task 5: Copy the Calculation to Count the Number of Processes**

Copy the INCREMENT COUNTER calculation from one of the other request definitions and paste it into the Calculations folder for the new CHECK PROCESS COUNT request definition.

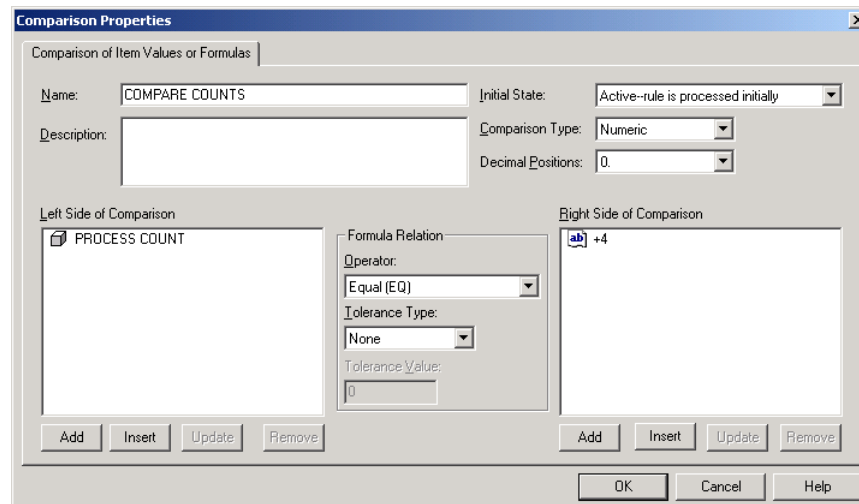
## Tutorial 4

### Validating the Process Flow

#### Task 6: Add a Comparison Rule to Check the Process Count

This final comparison rule will check to see if the four processes were completed.

1. Select the Comparisons folder under CHECK PROCESS COUNT, and then select **Edit > New**.
2. Select **Comparison of item values or formulas** as your **Rule Type**. Click **OK**.
3. Enter **COMPARE COUNTS** in the **Name** box. Use the defaults for Initial State.
4. Click in the window labeled **Left side of comparison**.
5. Click **Add** and select **PROCESS COUNT**. Click **OK**.
6. Click in the window labeled **Right side of comparison**.
7. Click **Add** and select a **Type** of **Numeric Literal Value**. A text box in which you can enter the literal is now available.
8. Enter +4 in the **Literal Value** box, select **(none)** from the **Operator** list, and click **OK**.
9. Verify that your final comparison looks like the following dialog box:



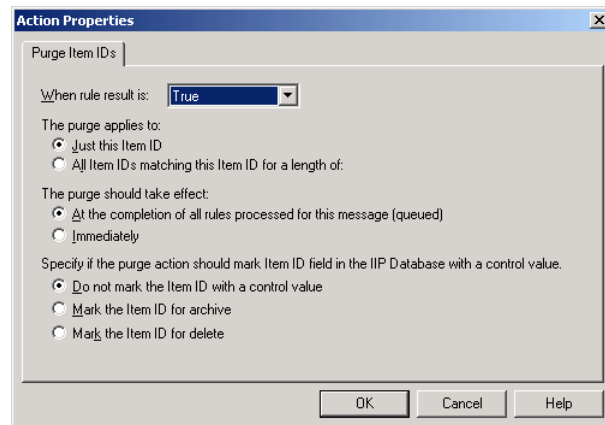
10. Click **OK** until you return to the ACR/Instream Editor main window.

#### Task 7: Add the Actions for the Comparison Rule

In this task you tell ACR/Instream what to do when the COMPARE COUNTS comparison completes its evaluation of process count.

1. Expand the Comparisons folder under CHECK PROCESS COUNT to display the **COMPARE COUNTS** comparison.

- Select the **COMPARE COUNTS** comparison and select **Edit > New > Purge Item IDs** to open the **Action Properties** dialog box.



The Purge Item IDs action removes one or more item IDs from ACR/Instream's memory.

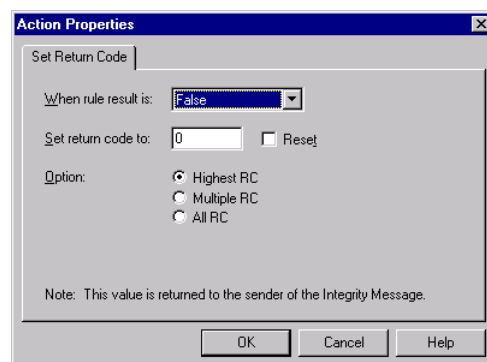
- Click **TRUE** for **When rule result is:**.
- Select **All Item IDs matching this item ID for a length of** and use the default value of 256 and click **OK**.

---

**Note:** When you reopen the properties box for this action, you will find that the **Just this item ID** is selected. This is correct when selecting a length of 256.

---

- Select the **COMPARE COUNTS** comparison again and select **Edit > New > Set Return Code** to open the **Action Properties** dialog box for the Set Return Code action.

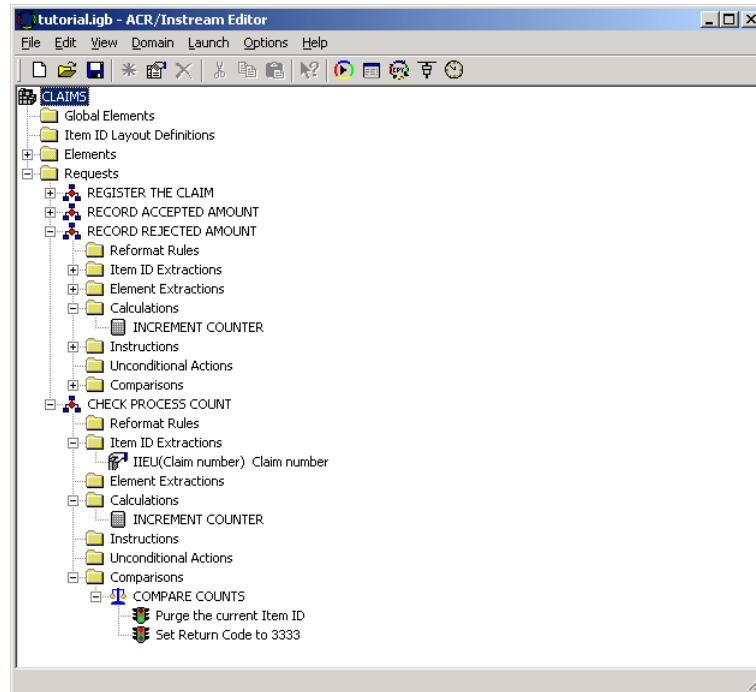


- Enter **3333** for **Set return code to** and click **OK**. Use the default of **FALSE** for "When rule result is:" and the **Highest RC** for the option choice.

## Tutorial 4

### Validating the Process Flow

7. Verify that your request definition looks like the following and then save your file.



### Task 8: Load and Test the Rules

Upload and test the rules using the procedures you learned in previous tutorials. Be sure to:

- Use **tutorial4.ieu** for your integrity message file
- Stop the ACR/Instream domain if you use the run65l command to load your rules
- Stop and start the ACR/Instream domain if you need to retest your rules, so you perform a cold start

The test file contains simulated problems. For this tutorial, the following problems should be detected:

Record number 11	Claim Number 031000	Return code of 2222	Fails balancing test: missing both messages for register the claim and accepted amount.
Record number 12	Claim Number 030003	Return code of 3333	Because the Reject Procedure is missing, the process count indicates a missing transaction.
Record number 19	Claim number 050005	Return code of 2222	Fails balancing test: either the amount accepted or the amount rejected is incorrect.
Record number 27	Claim number 061007	Return code of 3333	Because the Reject Procedure is missing, the process count indicates a missing transaction.

In Tutorial 2, you modified the startup file to perform a cold start. This enabled you to allow the data for each tutorial to be ignored for subsequent tasks. For normal operation, however, it's important to start ACR/Instream so that it remembers everything from before it was shut down.

The warm start enables ACR/Instream to load the status of all your business objects and pick up where it left off.

1. Open the file named runim.bat with any text editor, such as WordPad. The default location for this file is the following:

```
Program Files\Infogix\ACR Instream\Domain\runim.bat
```

2. Locate the following line with the COLDDDEBUG parameter:

```
SET PARMDAT=COLDDDEBUG
```

3. Change the parameter to WARMDEBUG:

```
SET PARMDAT=WARMDEBUG
```

4. Save the file. You must use the runim.bat name.

## Summary of the CLAIMS Item Group for Tutorial 4

This section describes the contents of the completed item group for tutorial 4. Components added in this tutorial are indicated with an asterisk (\*).

---

### Item Group: CLAIMS



**Elements:**

- BEGINNING AMOUNT (Extracted)
- ACCEPTED AMOUNT (Extracted)
- REJECTED AMOUNT (Extracted)
- TIME AT PROCESS 2 (Extracted)
- TIME AT PROCESS 3 (Extracted)
- \*PROCESS COUNT (Calculated)



#### Request Definition: REGISTER THE CLAIM

Purpose: Extracts data for BEGINNING AMOUNT  
\*Increments INCREMENT COUNTER calculation



#### Request Definition: RECORD ACCEPTED AMOUNT

Purpose: Extracts data for ACCEPTED AMOUNT and \*TIME AT PROCESS2  
\*Increments INCREMENT COUNTER calculation



#### Request Definition: RECORD REJECTED AMOUNT

Purpose: Extracts data for REJECTED AMOUNT and \*TIME AT PROCESS3  
Performs the INPUT-OUTPUT CHECK  
Checks for excessive time between processes  
\*Increments INCREMENT COUNTER calculation  
Instructions: "Investigate excessive processing time for claim"  
Comparison rule: INPUT-OUTPUT CHECK  
Formula: BEGINNING AMOUNT EQ ACCEPTED AMOUNT + REJECTED AMOUNT  
True action: Activate Rule comparison rule CHECK TIME  
False action: Set Return Code to 2222  
Comparison: CHECK TIME (activated by the INPUT-OUTPUT CHECK comparison)  
Formula: (TIME-AT-PROCESS-3 - TIME-AT-PROCESS-2) LT 2:30:00  
False action: Write Instruction  
False action: Produce Report  
False action: Set Return Code to 2525



#### \*Request Definition: CHECK PROCESS COUNT

Purpose: \*Increments INCREMENT COUNTER calculation  
\*Verifies claim completed four processes  
\*Comparison: COMPARE PROCESS COUNT TO REQUIRED COUNT  
\*Formula: PROCESS COUNT EQ 4  
\*True action: Purge Item ID  
False action: Set Return Code to 3333

---

# Glossary

## A

### **action**

A task performed by ACR/Instream based on the outcome of a comparison, such as launching a process or producing a report.

### **Activate Rule action**

An ACR/Instream action that changes an inactive comparison to active. This will cause the comparison to be evaluated at the end of the evaluation pass.

### **application program interface (API)**

A software routine that captures information about business objects, constructs an integrity message, and sends the message to ACR/Instream. For some installations, the API may also receive a message from ACR/Instream.

## B

### **business object**

A set of data items that can be uniquely identified, such as an insurance claim or a bank transaction. The set can be online (such as a credit card transaction) or a record in a batch file. Business objects typically have a blueprint or template and then data is uniquely filled in as needed.

### **business rules**

The comparisons and actions that will be applied to business object data. Business rules can be considered a specific subset of the more general term *rules*, which includes element extraction instructions, reformat rules, calculations, etc.

## C

### **calculation**

Calculations are values obtained by adding, subtracting, multiplying, or dividing element values, literals, or other calculated items.

### **client/server**

A relationship between two pieces of software that run independently of each other whereby one is a requestor of services (client) and the other is the provider (server).

### **cold start**

A function that restarts ACR/Instream using the initial program load procedure. A cold start deletes all existing items in memory since that last shutdown.

### **comparison**

A comparison performs the checking of values to determine the integrity of the target item's information. A comparison is part of a rule and consists of formulas and actions.

### **control architecture**

The combination of principles and, if required, design decisions that guide the practice of a traditional audit of an information system.

### **controls**

The collective rules that verify the information integrity of an application.

## D

### **data dictionary**

In the context of ACR/Instream, the data dictionary contains technical information about the layout definitions of the integrity message from the target application.

### **DEBUG**

An ACR/Instream function that turns debug tracing on and off. When tracing is on for a module, a trace printout is produced.

### **deferred actions**

Actions fall into one of three groups as to when they actually take effect. Actions can be taken at end-of-request, end-of-pass, or deferred. Deferred actions are sent to an ACR/Instream internal queue and processed after all end-of-pass and end-of-request actions have completed.

### **Dictionary Maker**

The ACR/Instream Dictionary Maker takes your COBOL copybook and creates an ACR/Instream Data Dictionary from it. If you are a COBOL user, this tool will significantly reduce the data dictionary development time and assure its accuracy.

### **Display Message action**

An ACR/Instream action that sends a message to either the console or a debug server.

### **domain**

A term that refers collectively to ACR/Instream's internal servers, definition files, and the domain manager.

### **domain global element**

A type of element that has a value available to all item groups. This is unlike a text or numeric element, which is associated with a single item ID.

### **domain manager**

The computer program that controls all processing within the domain. It is the recipient and dispatcher of integrity messages.

### **dummy item group (dummy IG)**

The dummy item group collects the names of request definitions that the domain has already encountered in a message, usually an integrity check message, but which have no rules in the rules definition file.

### **duplicate checking**

An ACR/Instream control type that detects a value in a specific transaction that is identical in length and content to the same value on a previous transaction.

### **duplicate checking element**

An element that works exclusively in a duplicate checking type comparison. This type of element holds the specific value for which ACR/Instream is checking for duplicates.

### **dynamic element**

Dynamic elements obtain their values by retrieving an element value from one or more items in the current or some other item group. In order to determine which items are to be selected and from which item group, the dynamic element rules use a search key.

### **dynamic element search key**

The set of characters that identify which items stored in ACR/Instream's memory are to be selected for the purpose of accumulating element values.

### **dynamic threshold**

An ACR/Instream control type that determines if the attributes of a transaction value are within an acceptable, dynamically calculated range.

## E

### **element**

An element is some form of data, such as data from your application. It can be extracted from the data portion of the integrity message or it can be a derived value, such as the result of a calculation.

### **element extraction**

Element extraction is the process of obtaining element values from an integrity message.

### **end-of-pass actions**

Actions fall into one of three groups as to when they actually take effect. Actions can be taken at end-of-request, end-of-pass, or deferred. End-of-pass actions are executed in the order they are specified in the rules. Unconditional end-of-pass actions are executed before the comparisons. Comparison end-of-pass actions are executed immediately after the comparison pass is completed. Comparison passes are made until no newly activated comparisons are encountered.

### **end-of-request actions**

Actions fall into one of three groups as to when they actually take effect. Actions can be taken at end-of-request, end-of-pass, or deferred. An end-of-request action is executed only once after all comparisons for that request definition are performed.

### **Execute Program action**

An ACR/Instream action that causes a user-written program to be executed. When executed, the program is passed two parameters, the integrity message and the contents of the target item, in that order.

**Extraction Error report**

A report that shows element names and values for any element that generated an extraction error.

**F****function**

A processing category within ACR/Instream.

**G****global element**

A type of element that is available to more than a single item group. See also *domain global element* and *item group global element*.

**H, I****IIEU**

Information integrity exchange unit (IIEU). As a prefix, this term identifies a component of an integrity message, such as IIEU-RC to identify the return code field. IIEU is also the unique identifier that begins each integrity message.

**Increment Element action**

An ACR/Instream action that causes the element value named in the rule to be incremented by 1.

**information integrity**

The accuracy, consistency, and reliability of the information content, process, and system.

**ACR/Instream**

A comprehensive software solution developed by Infogix that performs integrity validation for target applications that run continuously.

**ACR/Instream Data Dictionary Editor**

ACR/Instream software for entering technical information about layout definitions of the target application into the data dictionary.

**ACR/Instream Data Dictionary Maker**

The ACR/Instream Dictionary Maker takes your COBOL copybook and creates an ACR/Instream Data Dictionary from it. If you are a COBOL user, this tool will significantly reduce the data dictionary development time and assure its accuracy.

**ACR/Instream Editor**

ACR/Instream software that assists the user in the development of rules (request definitions, comparisons, and actions).

**ACR/Instream Player**

ACR/Instream software that assists the user in the development of business rules by testing the rules with real integrity messages.

**ACR/Instream Timer Editor**

ACR/Instream Timer Editor is the graphical user interface for the Timer utility. Using this editor, you can easily create timer sets and the messages to be executed with those timer sets.

**instructions**

Instructions are text you can specify to print on the Integrity Check report that provides users with directions for handling exceptions.

**INTEGRITY CHECK**

An ACR/Instream function that initiates the capture of business object information from an integrity message and executes a set of rules.

**Integrity Check report**

A report that shows the state of a controlled object and the results of the application for one set of rules to that object.

**integrity checkpoint**

The point at which the integrity of the business object is verified before the business object continues its life cycle.

**integrity message**

A set of data sent from the target application to ACR/Instream in the form of a single record. This record contains data about the application that can be used in your rules that detect errors. The message is sent by the application program interface (API).

**integrity validation**

The continuous, independent review of application processes to ensure the integrity of information. Integrity validation reduces the information uncertainty found in complex, real-time, continuously running systems.

### **item**

A specific instance of an item group. For example, for an item group of all insurance claims, a specific claim would be an item.

### **item group**

An item group is a template used by ACR/Instream to describe every individual member of that group. Define one item group for each business object you want to control.

### **item group global element**

A type of element whose value is available to all request definitions within the item group.

### **item ID**

A unique identifier for each item in an item group. The item ID for a controlled entity is typically extracted from the integrity message using the position and length specifications.

### **item ID extraction**

The process of extracting the item ID from an integrity message.

### **item key**

The total length of all extracted item IDs. This value controls the amount of storage needed for your business object. For example, if you have one item ID extracted for a length of 10 and another for a length of 5, the item key length for storage is 15. If you have one item ID, the item key length should be the same as the item ID extracted length.

### **Item Scan action**

An ACR/Instream action that applies rules from one request definition to all items in memory that match a generic search key. The items are sent to the internal ACR/Instream message queue with a new request definition.

### **item scan search key**

The set of characters that identify which items stored in ACR/Instream's memory are to be selected for further processing.

## **J**

### **JCL**

Job control language; a command language used for launching applications on the IBM OS/360 mainframe system.

## **K**

### **key**

One or more characters or perhaps a field within a data record used to identify the data and control its use.

## **L**

### **Launch Process action**

An ACR/Instream action that starts a process outside of ACR/Instream, such as sending email.

### **log file**

The log file is where ACR/Instream writes all integrity messages. The recovery log file is used for recovery after abnormal termination. The permanent log is used to recreate past executions.

## **M**

### **message layout**

The organization of data fields within an integrity message.

### **middleware**

Software which enables two or more application systems to communicate with each other when those systems are executing independent of each other.

For example, two systems executing on different platforms need middleware to communicate with each other. Similarly, two programs executing on the same platform but independently of each other need middleware to communicate.

### **Move action**

An ACR/Instream action that causes the results of a calculation to be moved into an element value.

## N

### node

A local area network device that communicates with other network devices.

## O, P

### process flow validation

An ACR/Instream control type that verifies that transactions are processed through the specified steps in the right order.

### Process Request ID action

An ACR/Instream action that causes the currently processed integrity message to be processed by another request definition after the current request definition is complete.

### Produce Report action

An ACR/Instream action that causes the Integrity Check report to be printed.

### purge

A deletion of an item ID from ACR/Instream's memory, effectively terminating control of the item by ACR/Instream.

### Purge Item IDs action

An ACR/Instream action that causes one or more items to be purged from ACR/Instream's memory. Which items are selected for purging depends on a search key that is specified as part of this action.

## Q

### queue

A data structure into which items are temporarily stored and then removed as needed.

ACR/Instream has four processing queues that store actions, process request IDs, external messages, and internal messages.

### Queued Request action

An ACR/Instream action that selects current data, formats it into a new integrity message, and sends it to the ACR/Instream internal queue so it can be processed by another request definition.

## R

### real-time

A description of computer processing systems that receive and process data quickly enough to produce output to control, direct, or affect the outcome of an ongoing activity or process.

### record layout

The organization of data fields within a record.

### reformat rule

A reformat rule acquires data from many sources (such as the item ID, literals, elements, etc.) to use with other ACR/Instream features, such as the Display Message action.

### Reject Message action

An ACR/Instream action that prevents the storing of the integrity message values for the target item. The effect of this action is to ignore or reject the values extracted from the integrity message by this request ID. Any values that existed for the target item before this request ID are left unchanged.

### request definition

A method to identify a set of rules to be used by ACR/Instream for processing an integrity message. The purpose of a request definition is to extract the item ID of the business object, extract data about the object, and apply comparisons to the data.

### request ID

The identifier of a specific request definition. The request ID is specified in the integrity message from the target application. ACR/Instream uses the request ID to select the request definition to apply to the data. The request ID must match the name of the request definition specified in the **Request Properties** dialog box.

### Reset Element Value action

An ACR/Instream action that resets the value of any element to its initial state.

### **rule**

An instruction interpreted by ACR/Instream to facilitate the detection of an information integrity error. A rule can instruct ACR/Instream how to identify the business object, capture relevant business object information from the integrity message, perform comparisons of the business object's captured information, and trigger actions based upon the outcomes of those comparisons.

### **rules file**

A file produced by the ACR/Instream Editor that is loaded into the rules definition file. This file normally has an extension of .igb.

## **S**

### **Set Return Code action**

An ACR/Instream action type that causes the return code in the returning integrity message to be set. As a result, when this action is triggered, a return code is sent back to the target application.

### **SHUTDOWN**

An ACR/Instream function that terminates the processing of ACR/Instream.

### **SWITCH FILES**

An ACR/Instream function that enables you to get a printout of the Integrity Check report by switching the Integrity Check report print output file. Essentially, this function toggles between two output files, allowing you to print the file that was just closed.

### **SYNCPOINT**

An ACR/Instream function that records in the syncpoint file the values of all active items currently in the element values table. These saved values are used for recovery purposes at startup after either a normal or an abnormal termination.

### **syncpoint file**

The syncpoint file contains the results of a SYNCPOINT function. The syncpoint file is an exact recording of the state of every controlled object at the time of the syncpoint. This file can be used to recreate ACR/Instream's internal element values table at system startup after a previous termination.

## **T**

### **target application**

The application to which ACR/Instream is applied.

### **target item**

The current specific instance of an item that is being acted upon by the controls.

### **timer set**

A timer set refers to the required messages that tell ACR/Instream when to execute a message and what function to execute. Specifically, a timer set is a CREATE TIMERS message and its associated ADD TIMER INFO messages.

### **Timer utility**

An ACR/Instream program that sends messages into the domain to be executed on a scheduled basis as if they were being sent from your application.

### **transaction**

A discrete activity within a computer system. If it has a unique identifier, it can be considered a transaction. Transactions are identified to ACR/Instream by their item ID.

### **transaction lifecycle tracking**

An ACR/Instream control type that detects late and lost transactions to ensure timely transaction receipt and processing.

## **U**

### **user interface**

The portion of a software program with which a user interacts. For example, the ACR/Instream Editor is a graphical user interface (GUI) for rules writing.

## **V**

### **value validation**

An ACR/Instream control type that monitors transaction values to ensure they balance at the end of the processing.

### W

#### **WARMSTART**

An ACR/Instream function that restarts ACR/Instream to the same state at which it was at the moment of its previous normal shutdown. If an abnormal termination occurs, a WARMSTART will initiate ACR/Instream's recovery process.

#### **Write Instruction action**

An ACR/Instream action that causes instructions to be printed on the Integrity Check report. If no print report action is executed during processing of this request ID, then this action will have no effect.