

Department of Veterans Affairs
eIV System Modifications
Technical Compliance Requirements
IB*2.0*506

Software Design Document (SDD)



August 2013

Version 1.1

Revision History

Date	Revision	Description	Author
7/18/13	1.0	Initial Version	FirstView Team
8/8/13	1.1	<p>Incorporated feedback from VA Product Development Review and CBO eBusiness eInsurance Team review.</p> <p>Dropped SRSRSD Requirement: 2.6.5.11</p> <p>Updated SRSRSD Requirement: 2.6.1.6 (required a design modification)</p> <p>Revised SRSRSD Requirement: 2.6.1.14 (design was originally written with the updated requirement in mind)</p> <p>Added SRSRSD Requirements: 2.6.1.20 through 2.6.1.22⁴ (added design)</p>	FirstView Team

Table of Contents

1	Introduction	1
1.1	Purpose	1
1.2	Scope.....	1
1.3	Inclusions.....	1
1.3.1	Scope Exclusions.....	2
1.4	Definitions and Acronyms.....	2
1.4.1	Acronyms/Abbreviations	2
1.4.2	Definitions	3
1.5	References	3
2	Conceptual Design.....	5
2.1	Product Perspective	5
2.1.1	User Interfaces.....	5
2.1.2	Hardware Interfaces.....	5
2.1.3	Software Interfaces	5
2.1.4	Communications Interfaces	5
2.1.5	Memory Constraints	5
2.1.6	Special Operations.....	5
2.2	Product Features	5
2.3	User Characteristics	5
2.4	Dependencies and Constraints	6
3	Specific Requirements.....	7
3.1	Database Repository	7
3.2	System Feature: Process Insurance Buffer.....	7
3.2.1	Functional Requirements:	7
3.2.2	Routines	8
3.2.3	Data Dictionaries.....	23
3.2.4	Protocols	24
3.3	System Feature: eIV HL7 Transactions	29
3.3.1	Functional Requirements:	29
3.3.2	Routines	30
3.3.3	Data Dictionaries.....	61
3.4	System Feature: Enhancements to the eIV Site Parameters.....	64
3.4.1	Functional Requirements:	64
3.4.2	Routines	64
3.4.3	Templates	66
3.5	System Feature: Enhancements using Security Keys	67
3.5.1	Functional Requirements:	67
3.5.2	Routines	69
3.5.3	Security Key.....	73
3.6	System Feature: Enhancements to Patient's Eligibility Benefits	74
3.6.1	Functional Requirements:	74
3.6.2	Routines	75
3.6.3	Data Dictionaries.....	92
4	Attachment A - Approval Signatures	94

1 Introduction

1.1 Purpose

The purpose of this Software Design Document (SDD) is to describe the changes needed to fulfill the requirements for the EDI - eInsurance System Modifications project.

The target audience for this SDD includes Office of Enterprise Development (OED), Product Development (PD), Product Support (PS), Software Quality Assurance (SQA), the Chief Business Office (CBO) Office and the software end-users.

1.2 Scope

The section describes the project scope which includes changes to the Veterans Health Information Systems and Technology Architecture (VistA) Integrated Billing (IB) module.

1.3 Inclusions

The following changes to the IB module are included in the project:

Enhancements to the Insurance Buffer

- Insurance Buffer – Display revised Insurance Buffer views
- Insurance Buffer – Remove the ability for a user to manually Verify a Buffer record
- Insurance Buffer - Rename the VistA file that stores the records for the Insurance Buffer
- Insurance Buffer – Disable the ability to create new Insurance companies and Group/Plans
- Insurance Buffer – Create Buffer entry for those appointments that meet the extract criteria but the payer is Nationally Inactive

eIV HL7 Transactions

- HL7 Transactions - Modify the daily eIV registration message (HL7 MFN^M01 message) that is sent to the Financial Service Center (FSC) in Austin, TX to include additional data
- HL7 Transactions – Update/Create several eIV Site Parameters in Vista using the existing table update message (HL7 MFN^M01 message)
- HL7 Transactions – Modify VistA's retry methodology (resending inquiries to the Financial Service Center (FSC) in Austin, TX)

Enhancements to the eIV Site Parameters

- eIV Site Parameters - Enhancements to several existing General Parameters (Timeout Days, Freshness Days, & HL7 Response Processing)
- eIV Site Parameters – Add a new General Parameter associated with Retries (resending inquiries to the Financial Service Center (FSC) in Austin, TX)

Enhancements using Security Keys

- Security Keys – Define new security keys to control add/edits of insurance companies and group/plans

Enhancements to Patient's Eligibility Benefits

- Eligibility Benefits - Enhancements to update the eligibility benefits viewed from the Process Insurance Buffer option
- Eligibility Benefits - Enhancements to update the eligibility benefits viewed from the Patient Insurance Info View/Edit option
- Eligibility Benefits - Enhancements to update the eligibility benefits viewed from the TPJI option
- Eligibility Benefits - Enhancements to add eligibility benefit information to the existing eIV Response Report

*Modifications to HL7 Transactions are dependent on approval of HL7 group.

1.3.1 Scope Exclusions

The Insurance Capture Buffer (ICB) module currently extracts data from the VistA IB database to enhance insurance data collection and verification processes for VA facilities. The enhancements in this project will provide additional data that could be included in the ICB extract; however, modification of the ICB extract is beyond the scope of this project.

1.4 Definitions and Acronyms

1.4.1 Acronyms/Abbreviations

Acronyms	Description
270 Transaction	Eligibility Benefit Inquiry – ASC X12N 5010 Health Care Eligibility Benefit Inquiry and Response
271 Transaction	Eligibility Benefit Response - ASC X12N 5010 Health Care Eligibility Benefit Inquiry and Response
AAC	Austin Automation Center
ASC X12N	Accredited Standards Committee X12N
CBO	Chief Business Office
CM	Configuration Management
CORE	Committee on Operating Rules for Information Exchange
CR	Change Request
DMI	Data Management Interface
EC	Eligibility Communicator
EDI	Electronic Data Interchange
eIV	The term used to described projects related to the 270/271 transactions
FSC	Financial Services Center – Austin, Texas
HIPAA	Health Insurance Portability and Accountability Act of 1996
HL7	Health Level 7
IB	Integrated Billing software version 2.0
ICB	Insurance Capture Buffer


Acronyms	Description
IRM	Information Resources Management
MFN	Master File Notification - HL7 Message
M (MUMPS)	Massachusetts General Hospital Utility Multi-Programming System
OED	Office of Enterprise Development
PD	Product Development
PS	Product Support
RSD	Requirements Specification Document
SACC	Standards and Conventions Committee
TSPR	Technical Services Project Repository
VistA	Veterans Health Information Systems and Technology Architecture

1.4.2 Definitions

Term	Definition
270	EDI transaction set for Health Care Eligibility Benefit Inquiry, used to send an inquiry to a trading partner - ASC X12N 5010 Health Care Eligibility Benefit Inquiry and Response
271	EDI transaction set for Health Care Eligibility Benefit Response This is returned from the payer to the billing facility - ASC X12N 5010 Health Care Eligibility Benefit Inquiry and Response
HL7	A framework (and related standards) for the exchange, integration, sharing and retrieval of electronic health information
ICB	ICB module is an insurance card scanning and VistA buffer file update management system designed to enhance insurance data collection and verification processes for VA facilities including medical centers and Community Based Outpatient Clinic(CBOC)s
Payer	An insurance company, fiscal intermediary, government agency, other agency, or individual responsible for the payment of health care claims
User	The person or persons who operate or interact directly with this product.

1.5 References

Name	Location	Date
Electronic Insurance Verification User Guide		January 2012
Electronic Insurance Verification Technical Manual/Security Guide		August 2011

Name	Location	Date
Requirements Specification Document eIV System Modifications Technical Compliance Requirements v1.2	TSPR – Project: eInsurance System Modifications 	8/8/13

2 Conceptual Design

2.1 Product Perspective

2.1.1 User Interfaces

Users will use the existing Integrated Billing software using the roll and scroll interface.

2.1.2 Hardware Interfaces

The VistA software runs on the standard hardware platforms used by the Department of Veterans Affairs Healthcare facilities. These systems consist of Alpha systems running Open VMS and InterSystems Cache. There is no special hardware interface associated with this effort.

2.1.3 Software Interfaces

The Integrated Billing package interfaces with multiple VistA packages, as well as DMI to AAC.

2.1.4 Communications Interfaces

The eIV software uses HL7 messages to communicate with the Financial Services Center (FSC).

2.1.5 Memory Constraints

Overall impact on VistA resources is low on system overhead and CPU, as well as on data storage and network usage.

2.1.6 Special Operations

No special operations, other than the standard VistA operations for backup, recovery or archiving will be needed.

2.2 Product Features

The EDI – eIV Systems Modifications project will include changes to the following features:

- Insurance Buffer
- eIV HL7 Transactions
- eIV Site Parameters
- Security Keys
- Patient's Eligibility Benefits

2.3 User Characteristics

Representatives at VA Medical Centers, including Billing Clerks, Billing Supervisors, Insurance Verifiers, and IRM staff will use the functionality provided by this project.

2.4 Dependencies and Constraints

The following constraints will affect this project:

- The HL7 Version 2.4 standards;
- The ASC X12N 270 and ASC X12N 271 standards;
- The Health Insurance Portability and Accountability Act (HIPAA);
- Payer's X12 271 responses via clearing house and FSC.

3 Specific Requirements

For a high level summary of the features and modifications that will be made to the IB module as part of this project refer to Section 1.3 Inclusions.

3.1 Database Repository

The data for this project will be stored locally in the VistA database (M globals).

3.2 System Feature: Process Insurance Buffer

This section covers the design for the requirements listed in Section 2.6.1 of the RSD.

3.2.1 Functional Requirements:

Table 1: Enhancements to the Insurance Buffer

SRSRSD REQ ID	REQ Title	Comments / Notes
2.6.1.1	Insurance Buffer – Create New ‘Complete Buffer’ (CB) Screen	
2.6.1.2	Insurance Buffer – Default View to be the ‘Complete Buffer’ (CB) Screen	
2.6.1.3	Insurance Buffer – ‘Complete Buffer’ Screen Contents	
2.6.1.4	Insurance Buffer – ‘Complete Buffer’ Screen Actions	Actions are standard across screens – no code change needed
2.6.1.5	Insurance Buffer – Ability to Jump to the ‘Complete Buffer’ Screen	
2.6.1.6	Insurance Buffer – ‘Positive Buffer’ Screen Fix Filter	
2.6.1.7	Insurance Buffer – ‘Medicare Buffer’ Screen Fix Filter	Currently prints all Medicare records – no code change needed
2.6.1.8	Insurance Buffer – ‘Negative Buffer’ Screen Fix Filter	
2.6.1.9	Insurance Buffer – Remove ‘Future Appointments’ Screen	
2.6.1.10	Insurance Buffer – Create New ‘Failure Buffer’ (FB) Screen	
2.6.1.11	Insurance Buffer – ‘Failure Buffer’ Screen Contents	
2.6.1.12	Insurance Buffer – ‘Failure Buffer’ Screen Actions	Actions are standard across screens – no code change needed
2.6.1.13	Insurance Buffer – Remove ‘Verify Entry’ Action from the Buffer Views	

<u>SRSRSD</u> REQ ID	REQ Title	Comments / Notes
2.6.1.14	Insurance Buffer – Filter Insurance Buffer Records Based on User's Security Keys (REVISED)	Revised on 8/8/13 and is referenced in the <u>SRSRSD</u> version 1.2
2.6.1.15	Insurance Buffer – Ability to Jump to the 'Failure Buffer' Screen	
2.6.1.16	Insurance Buffer – Remove Ability to Create New Insurance Company	Addressed in section 3.5.2 of this SDD.
2.6.1.17	Insurance Buffer – Remove Ability to Create New Group/Plan	Addressed in section 3.5.2 of this SDD.
2.6.1.18	Insurance Buffer – Rename the Insurance Buffer File	
2.6.1.19	Insurance Buffer – Create Insurance Buffer Entry for Appointments with Nationally Inactive Payers	
2.6.1.20	Insurance Buffer – Add "Escalate" Action to the Buffer Views	New requirement referenced in the <u>SRSRSD</u> version 1.2
2.6.1.21	Insurance Buffer – Restrict use of the "Escalate" Action	New requirement referenced in the <u>SRSRSD</u> version 1.2
2.6.1.22	Insurance Buffer – Implement the "Escalate" Action	New requirement referenced in the <u>SRSRSD</u> version 1.2

3.2.2 Routines

Routine Name	IBCNBLL			
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change			
Requirement Traceability Matrix	2.6.1.1, 2.6.1.2, 2.6.1.3, 2.6.1.6, 2.6.1.8, 2.6.1.14			
Related Options	"Complete Buffer", "Positive Buffer", "Negative Buffer" screens for Insurance Buffer Lists			
Related Routines	Routines "Called By"	Routines "Called"		
		ALL^IBCNS1 INSERROR^IBCNEUT3		
Data Dictionary (DD) References				
Related Protocols				
Related Integration Control Registrations (ICRs)				
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local			

Routine Name	IBCNBLL
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
<pre> EN ; - main entry point for screen N VIEW,AVIEW,DFLG S VIEW=1,AVIEW=0 ; default to positive view K ^TMP("IBCNERTQ",\$J) ; clear temp. global for eIV real time inquiries D EN^VALM("IBCNB INSURANCE BUFFER LIST") Q ; HDR ; header code for list manager display S VALMHDR(1)="Sorted by: "_\$P(IBCNSORT,U,2) I \$P(IBCNSORT,U,3)'="" S VALMHDR(1)=VALMHDR(1)_" , ""_\$P(IBCNSORT,U,3)_" "" first" I VIEW=1 S VALM("TITLE")="Positive Insurance Buffer",VALMSG="**Verified +Active ?Await/Reply" I VIEW=2 S VALM("TITLE")="Negative Insurance Buffer",VALMSG="**Verified -N/Active #Unclear !Unable/Send" I VIEW=3 S VALM("TITLE")="Medicare(WNR) Insurance Buffer",VALMSG="**Verified +Act -N/Act ?Await/R #Unclr !Unable/Send" I VIEW=4 S VALM("TITLE")="Future Appointments Buffer",VALMSG="!Unable/Send" I VIEW=5 S VALM("TITLE")="e-Pharmacy Buffer",VALMSG="**Verified" ; IB*2*435 Q ; BLD ; build screen display N IBCNT,IBCNS1,IBCNS2,IBBUFDA,IBLINE ; D SORT S IBCNT=0,VALMCNT=0,IBBUFDA=0 ; S IBCNS1="" F S IBCNS1=\$O(^TMP(\$J,"IBCNBLLS",IBCNS1)) Q:IBCNS1="" D .S IBCNS2="" F S IBCNS2=\$O(^TMP(\$J,"IBCNBLLS",IBCNS1,IBCNS2)) Q:IBCNS2="" D ..S IBBUFDA=0 F S IBBUFDA=\$O(^TMP(\$J,"IBCNBLLS",IBCNS1,IBCNS2,IBBUFDA)) Q:'IBBUFDA D ...S DFLG=^TMP(\$J,"IBCNBLLS",IBCNS1,IBCNS2,IBBUFDA) ...S IBCNT=IBCNT+1 I '\$D(ZTQUEUEU),'(IBCNT#15) W " ." ...S IBLINE=\$\$BLDLN(IBBUFDA,IBCNT,DFLG) ...D SET(IBLINE,IBCNT) ; I VALMCNT=0 D SET("",0),SET("There are no Buffer entries that have not been processed.",0) Q ; BLDLN(IBBUFDA,IBCNT,DFLG) ; build line to display on List screen for one Buffer entry N DFN,IB0,IB20,IB60,IBLINE,IBY,VAIN,VADM,VA,VAERR,X,Y,IBMTS S IBLINE="",IBBUFDA=+\$G(IBBUFDA) S IB0=\$G(^IBA(355.33,IBBUFDA,0)),IB20=\$G(^IBA(355.33,IBBUFDA,20)),IB60=\$G(^IBA(355.33,IBBUFDA </pre>	

Routine Name	IBCNBLL
<pre> ,60)) S DFN=+IB60 I +DFN D DEM^VADPT,INP^VADPT ; S IBY=\$G(IBCNT),IBLINE=\$\$SETSTR^VALM1(IBY,"",1,4) ; ; ESG - 6/6/02 - SDD 5.1.8 ; pull the symbol from the symbol function ; S IBY=\$\$SYMBOL(IBBUFDA) S IBY=IBY_\$P(\$G(^DPT(+DFN,0)),U,1),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,5,20) S IBLINE=\$\$SETSTR^VALM1(DFLG,IBLINE,25,1) S IBY=\$G(VA("BID")),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,27,4) S IBY=\$P(IB20,U,1),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,32,17) S IBY=\$P(IB60,U,4),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,50,13) S IBY=\$\$GET1^DIQ(355.12,\$P(IB0,U,3),.03),IBLINE=\$\$SETSTR^VALM1(\$\$SRCCNV(IBY),IBLINE,64,1) S IBY=\$\$DATE(+IB0),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,66,8) S IBY="" D S IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,76,5) . S IBY=IBY_\$S(+\$\$INSURED^IBCNS1(DFN,DT):"i",1:" ") . S IBY=IBY_\$S(+\$G(VAIN(1)):"I",1:" ") . S IBY=IBY_\$S(+\$G(VADM(6)):"E",1:" ") . S IBMTS=\$P(\$\$LST^DGMTU(DFN),U,4) . S IBY=IBY_\$S(IBMTS="C":"Y",IBMTS="G":"Y",1:" ") . S IBY=IBY_\$S(+\$\$HOLD(DFN):"H",1:" ") Q IBLINE ; SORT ; set up sort for list screen ; 1^Patient Name, 2^Ins Name, 3^Source Of Info, 4^Date Entered, 5^Inpatient (Y/N), 6^Means Test (Y/N), 7^On Hold, 8^Verified, 9^eIV Status, 10^Positive Response N APPTNUM,IB0,IB20,IB60,IBCNDT,IBBUFDA,IBCNDFN,IBCNPAT,IBCSORT1,IBCSORT2,IBSDA,DFN ,VAIN,VA,VAERR,IBX,IBCNT,INAME,SYM,MWNRFLG,MWNRLEN,X,Y S IBCNT=0 ; K ^TMP(\$J,"IBCNBLLS") I '\$G(IBCNSORT) S IBCNSORT="1^Patient Name" ; get payer ien for Medicare WNR S MWNRLEN=\$P(\$G(^IBE(350.9,1,51)),U,25) ; S IBCNDT=0 F S IBCNDT=\$O(^IBA(355.33,"AEST","E",IBCNDT)) Q:'IBCNDT D .S IBBUFDA=0 F S IBBUFDA=\$O(^IBA(355.33,"AEST","E",IBCNDT,IBBUFDA)) Q:'IBBUFDA D ..S IBCNT=IBCNT+1 I '\$D(ZTQUEUED),'(IBCNT#15) W "." ..S IB0=\$G(^IBA(355.33,IBBUFDA,0)),IB20=\$G(^IBA(355.33,IBBUFDA,20)),IB60=\$G(^IBA(355.33,IBBU FDA,60)) ..S IBCNDFN=+IB60,IBCNPAT="" I +IBCNDFN S IBCNPAT=\$P(\$G(^DPT(IBCNDFN,0)),U,1) ..S INAME=\$P(IB20,U) ..; ..I +IBCNSORT=1 S IBCSORT1=IBCNPAT ..I +IBCNSORT=2 S IBCSORT1=INAME </pre>	

Routine Name	IBCNBLL
<pre> ..I +IBCN SORT=3 S IBCSORT1=\$P(IB0,U,3) ..I +IBCN SORT=4 S IBCSORT1=\$P(+IB0,".",1) ..I +IBCN SORT=5 I +IBCND FN S DFN=+IBCND FN D INP^VADPT S IBCSORT1=\$S(\$G(VAIN(1)):1,1:2) ..I +IBCN SORT=6 I +IBCND FN S IBX=\$P(\$LST^DGMTU(IBCND FN),U,4) S IBCSORT1=\$S(IBX="C":1,IBX="G":1,1:2) ..I +IBCN SORT=7 I +IBCND FN S IBX=\$\$HOLD(IBCND FN) S IBCSORT1=\$S(+IBX:1,1:2) ..I +IBCN SORT=8 S IBCSORT1=\$S(+P(IB0,U,10):1,1:2) ..; Sort by symbol and then within the symbol, sort by date entered ..; Build a numerical subscript with format ##.FM date ..S SYM=\$\$SYMBOL(IBBUFDA) ..I +IBCN SORT=9 S IBCSORT1=\$G(IBCNSORT(1,SYM))_"_"\$P(+IB0,".",1),IBCSORT1=+IBCSORT1 ..; ..I +IBCN SORT=10 S IBCSORT1=\$S(SYM="+":0,1:1),IBCSORT2=IBCNPAT ..; ..S IBCSORT1=\$S(\$G(IBCNSORT1)=""~UNKNOWN",1:IBCSORT1),IBCSORT2=\$S(IBCNPAT=""~UNK NOWN",1:IBCNPAT) ..; get future appointments ..S IBSDA(1)=DT,IBSDA(3)="R;l;NT",IBSDA(4)=IBCND FN,IBSDA("FLDS")="1;2" ..S DFLG="",APPTNUM=\$\$SDAPI^SDAMA301(.IBSDA) I APPTNUM>0,SYM="!" S DFLG="d" ; duplicate flag ..S MWN RFLG=0 I MWNRIEN="",P(\$P(\$\$INSERROR^IBCNEUT3("B",IBBUFDA),U,2)=MWNRIEN S MWN RFLG=1 ..I VIEW=1 Q:MWN RFLG=1 Q:SYM="*"&(SYM="+")&(SYM="?")&(SYM=" ") ..I VIEW=2 Q:MWN RFLG=1 Q:SYM="*"&(SYM="-")&(SYM="#")&(SYM="!") ..I VIEW=3 Q:MWN RFLG=0 ..I VIEW=4 Q:SYM="!" Q:APPTNUM<1 M ^TMP(\$J,"IBCNAPPTS")=^TMP(\$J,"SDAMA301") ..I VIEW=5 Q:\$P(IB0,U,17) ; IB*2*435 e-Pharmacy view only ..I VIEW=5 Q:\$P(IB0,U,17) ; IB*2*435 ..S ^TMP(\$J,"IBCNBLLS",IBCSORT1,IBCSORT2,IBBUFDA)=DFLG ..K VAIN,IBCSORT1,IBCSORT2 ..Q .Q I IBCNT,\$D(ZTQUEUED) W " " Q ; </pre>	
Modified Logic (Changes are in bold)	
<pre> EN ; - main entry point for screen N VIEW,AVIEW,DFLG ;IB*2.0*506 Default view changed to Complete View S VIEW=6,AVIEW=0 K ^TMP("IBCNERTQ",\$J) ; clear temp. global for eIV real time inquiries D EN^VALM("IBCNB INSURANCE BUFFER LIST") Q ; HDR ; header code for list manager display S VALMHDR(1)="Sorted by: " _\$P(IBCNSORT,U,2) I \$P(IBCNSORT,U,3)'="" S VALMHDR(1)=VALMHDR(1)_"", "" _\$P(IBCNSORT,U,3)_" "" first" ;IB*2.0*506 Only Verified and Active records. I VIEW=1 S VALM("TITLE")="Positive Insurance Buffer",VALMSG="*Verified +Active" ;IB*2.0*506 Only Verified and N/Active records. I VIEW=2 S VALM("TITLE")="Negative Insurance Buffer",VALMSG="*Verified -N/Active" </pre>	

Routine Name	IBCNBLL
<pre> I VIEW=3 S VALM("TITLE")="Medicare(WNR) Insurance Buffer",VALMSG="*Verified +Act -N/Act ?Await/R #Unclr !Unable/Send" I VIEW=4 S VALM("TITLE")="Future Appointments Buffer",VALMSG="!Unable/Send" I VIEW=5 S VALM("TITLE")="e-Pharmacy Buffer",VALMSG="*Verified" ; IB*2*435 ;IB*2.0*506 Added Complete Buffer View I VIEW=6 S VALM("TITLE")="Complete Buffer",VALMSG="" I VIEW=7 S VALM("TITLE")="Failure Buffer",VALMSG="!Unable/Send" Q ; INIT ; initialization for list manager list K ^TMP("IBCNBLL",\$J),^TMP("IBCNBLLX",\$J),^TMP("IBCNBLLY",\$J),^TMP(\$J,"IBCNBLLS"),^ TMP(\$J,"IBCNAPPTS") S:\$G(IBCNSORT)="" IBCNSORT=\$S(VIEW=1:"10^Positive Response",1:"1^Patient Name") S IBKEYS=\$\$GETKEYS(DUZ) D BLD Q ; GETKEYS(DUZ) ; ;Make sure that user has the INSURANCE EDIT key and/or the GROUP/PLAN EDIT key. User ;must have either key in order to see non_Positive Entries. N KEY1,KEY2 S KEY1=\$O(^DIC(19.1,"E","INSURANCE EDIT",)) I KEY1="" S KEY1=\$D(^VA(200,DUZ,51,KEY1)) S KEY2=\$O(^DIC(19.1,"E","GROUP/PLAN EDIT",)) I KEY2="" S KEY1=\$D(^VA(200,DUZ,51,KEY1)) Q KEY1!KEY2 ; BLD ; build screen display N IBCNT,IBCNS1,IBCNS2,IBBUFDA,IBLINE ; D SORT S IBCNT=0,VALMCNT=0,IBBUFDA=0 ; S IBCNS1="" F S IBCNS1=\$O(^TMP(\$J,"IBCNBLLS",IBCNS1)) Q:IBCNS1="" D .S IBCNS2="" F S IBCNS2=\$O(^TMP(\$J,"IBCNBLLS",IBCNS1,IBCNS2)) Q:IBCNS2="" D ..S IBBUFDA=0 F S IBBUFDA=\$O(^TMP(\$J,"IBCNBLLS",IBCNS1,IBCNS2,IBBUFDA)) Q:IBBUFDA D ...S DFLG=^TMP(\$J,"IBCNBLLS",IBCNS1,IBCNS2,IBBUFDA) ...S IBCNT=IBCNT+1 I '\$D(ZTQUEUED),'(IBCNT#15) W ". " ...S IBLINE=\$\$BLDLN(IBBUFDA,IBCNT,DFLG) I IBLINE="" Q ...D SET(IBLINE,IBCNT) ; I VALMCNT=0 D SET("",0),SET("There are no Buffer entries that have not been processed.",0) Q ; BLDLN(IBBUFDA,IBCNT,DFLG) ; build line to display on List screen for one Buffer entry </pre>	

Routine Name	IBCNBLL
<pre> N DFN,IB0,IB20,IB60,IBLINE,IBY,VAIN,VADM,VA,VAERR,X,Y,IBMTS,IBINSCO S IBLINE="",IBBUFDA=+\$G(IBBUFDA) S IB0=\$G(^IBA(355.33,IBBUFDA,0)),IB20=\$G(^IBA(355.33,IBBUFDA,20)),IB60=\$G(^IBA(355.33,I BBUFDA,60)) S DFN=+IB60 I +DFN D DEM^VADPT,INP^VADPT ; ;IB*2.0*506 – If IBKEYS only allow active insurance companies. I 'IBKEYS,'\$\$ACTIVE(DFN,IB20) G BLDLNQ ; S IBY=\$G(IBCNT),IBLINE=\$\$SETSTR^VALM1(IBY,"",1,4) ; ; ESG - 6/6/02 - SDD 5.1.8 ; pull the symbol from the symbol function ; S IBY=\$\$SYMBOL(IBBUFDA) S IBY=IBY_\$P(\$G(^DPT(+DFN,0)),U,1),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,5,20) S IBLINE=\$\$SETSTR^VALM1(DFLG,IBLINE,25,1) S IBY=\$G(VA("BID")),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,27,4) S IBY=\$P(IB20,U,1),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,32,17) S IBY=\$P(IB60,U,4),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,50,13) S IBY=\$\$GET1^DIQ(355.12,\$P(IB0,U,3),.03),IBLINE=\$\$SETSTR^VALM1(\$\$SRCCNV(IBY),IBLIN E,64,1) S IBY=\$\$DATE(+IB0),IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,66,8) S IBY="" D S IBLINE=\$\$SETSTR^VALM1(IBY,IBLINE,76,5) . S IBY=IBY_\$\$(+\$\$INSURED^IBCNS1(DFN,DT):"I",1:" ") . S IBY=IBY_\$\$(+\$G(VAIN(1)):"I",1:" ") . S IBY=IBY_\$\$(+\$G(VADM(6)):"E",1:" ") . S IBMTS=\$P(\$\$LST^DGMTU(DFN),U,4) . S IBY=IBY_\$\$(IBMTS="C": "Y",IBMTS="G": "Y",1:" ") . S IBY=IBY_\$\$(+\$\$HOLD(DFN):"H",1:" ") BLDLNQ ; Q IBLINE ; ACTIVE(DFN,IB20) ;Check for active insurance N IBACT,IBINSCO,IBINSNM S IBINSNM=\$P(IB20,U,1) S IBACT=0 D ALL^IBCNS1(DFN,IBINSCO,1,DT,0) </pre>	

Routine Name	IBCNBLL
<pre> F X=1:1:IBINSCO(0) D I IBACT Q . I \$P(IBINSCO(X,0),U,1)=IBINSNM S IBACT=1 ACTIVEQ ; Q IBACT ; SORT ; set up sort for list screen ; 1^Patient Name, 2^Ins Name, 3^Source Of Info, 4^Date Entered, 5^Inpatient (Y/N), 6^Means Test (Y/N), 7^On Hold, 8^Verified, 9^eIV Status, 10^Positive Response N APPTNUM,IB0,IB20,IB60,IBCNDDT,IBBUFDA,IBCNDFN,IBCNPAT,IBCSORT1,IBCSORT2,IB SDA,DFN,VAIN,VA,VAERR,IBX,IBCNT,INAME,SYM,MWNRFLG,MWNRLEN,X,Y S IBCNT=0 ; K ^TMP(\$J,"IBCNBLLS") I '\$G(IBCNSORT) S IBCNSORT="1^Patient Name" ; get payer ien for Medicare WNR S MWNRLEN=\$P(\$G(^IBE(350.9,1,51)),U,25) ; S IBCNDDT=0 F S IBCNDDT=\$O(^IBA(355.33,"AEST","E",IBCNDDT)) Q:'IBCNDDT D .S IBBUFDA=0 F S IBBUFDA=\$O(^IBA(355.33,"AEST","E",IBCNDDT,IBBUFDA)) Q:'IBBUFDA D ..S IBCNT=IBCNT+1 I '\$D(ZTQUEUED),'(IBCNT#15) W "." ..S IB0=\$G(^IBA(355.33,IBBUFDA,0)),IB20=\$G(^IBA(355.33,IBBUFDA,20)),IB60=\$G(^IBA(355 .33,IBBUFDA,60)) ..S IBCNDFN=+IB60,IBCNPAT="" I +IBCNDFN S IBCNPAT=\$P(\$G(^DPT(IBCNDNFN,0)),U,1) ..S INAME=\$P(IB20,U) ..; ..I +IBCNNSORT=1 S IBCNSORT1=IBCNPAT ..I +IBCNNSORT=2 S IBCNSORT1=INAME ..I +IBCNNSORT=3 S IBCNSORT1=\$P(IB0,U,3) ..I +IBCNNSORT=4 S IBCNSORT1=\$P(+IB0,".",1) ..I +IBCNNSORT=5 I +IBCNDFN S DFN=+IBCNDFN D INP^VADPT S IBCNSORT1=\$S(\$G(VAIN(1)):1,1:2) ..I +IBCNNSORT=6 I +IBCNDFN S IBX=\$P(\$\$LST^DGMTU(IBCNDNFN),U,4) S IBCNSORT1=\$S(IBX="C":1,IBX="G":1,1:2) ..I +IBCNNSORT=7 I +IBCNDFN S IBX=\$\$HOLD(IBCNDNFN) S IBCNSORT1=\$S(+IBX:1,1:2) ..I +IBCNNSORT=8 S IBCNSORT1=\$S(+P(IB0,U,10):1,1:2) ..; Sort by symbol and then within the symbol, sort by date entered ..; Build a numerical subscript with format ##.FM date ..S SYM=\$\$SYMBOL(IBBUFDA) ..I +IBCNNSORT=9 S IBCNSORT1=\$G(IBCNSORT(1,SYM))_"_"\$P(+IB0,".",1),IBCSORT1=+IBCSORT1 ..; ..I +IBCNNSORT=10 S IBCNSORT1=\$S(SYM="+":0,1:1),IBCSORT2=IBCNPAT ..; ..S IBCNSORT1=\$S(\$G(IBCNSORT1)=""~UNKNOWN",1:IBCSORT1),IBCSORT2=\$S(IBCNPAT =""~UNKNOWN",1:IBCNPAT) ..; get future appointments ..S IBSDA(1)=DT,IBSDA(3)="R;I;NT",IBSDA(4)=IBCNDFN,IBSDA("FLDS")="1;2" </pre>	

Routine Name	IBCNBLL
<pre> ..S DFLG="",APPTNUM=\$\$SDAPI^SDAMA301(.IBSDA) I APPTNUM>0,SYM="!" S DFLG="d" ; duplicate flag ..S MWNRFLG=0 I MWNRIEN="", \$P(\$\$INSERTOR^IBCNEUT3("B",IBBUFDA),U,2)=MWNRIEN S MWNRFLG=1 ..;IB*2.0*506 Only allow * and + symbols for Positive Buffer ..I VIEW=1 Q:MWNRFLG=1 Q:SYM="*"&(SYM="+")&(SYM="\$") ..;IB*2.0*506 Only Allow * and – symbols for Negative Buffer ..I VIEW=2 Q:MWNRFLG=1 Q:SYM="*"&(SYM="-") ..I VIEW=3 Q:MWNRFLG=0 ..I VIEW=4 Q:SYM="!" Q:APPTNUM<1 M ^TMP(\$J,"IBCNAPPTS")=^TMP(\$J,"SDAMA301") ..;IB*2.0*506 Include e-Pharmacy on Complete View ..I VIEW=5!(VIEW=6) Q:\$P(IB0,U,17) ; IB*2*435 e-Pharmacy view only ..I VIEW=5&(VIEW=6) Q:\$P(IB0,U,17) ; IB*2*435 ..I VIEW=7 Q:MWNRFLG=1 Q:SYM="!" ..S ^TMP(\$J,"IBCNBLLS",IBCSORT1,IBCSORT2,IBBUFDA)=DFLG ..K VAIN,IBCSORT1,IBCSORT2 ..Q ..Q I IBCNT,'\$D(ZTQUEUED) W "!" Q ; </pre>	

Routine Name	EN^IBCNEDE2	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.1.19	
Related Options	“Future Appointments” screen	
Related Routines	Routines “Called By”	Routines “Called”
	^IBCNEDE	SETTINGS^IBCNEDE7 ALL^IBCNS1 EXCLUDE^IBCNEUT4 INSERTOR^IBCNEUT3 BFEXIST^IBCNEUT5 UPDDTS^IBCNEDE6 TQUPDSV^IBCNEUT5 ADDTQ^IBCNEUT5 SIDCHK^IBCNEDE5 PT^IBCNEBF

Routine Name	EN^IBCNEDE2
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
<pre> ; EN ; Loop through designated cross-references for updates ; Pre reg extract (Appointment extract) ; N TODAYSDT,FRESHDAY,SLCCRIT1,MAXCNT,CNT,ENDDT,CLNC,FRESHDT,GIEN N APTDT,INREC,INSIEN,PAYER,PIEN,PAYERSTR,SYMBOL,SUPPBUFF,PATID N DFN,OK,VAIN,INS,DATA1,DATA2,ELG,PAYERID,SETSTR,SRVICEDT,ACTINS N TQIEN,IBINDT,IBOUTP,QUERYFLAG,INSNAME,FOUND1,FOUND2,IBCNETOT,VDATE N SID,SIDACT,SIDDATA,SIDARRAY,SIDCNT,IBDDI,IBINS,DISYS,NUM,MCAREFLG ; S SETSTR=\$\$SETTINGS^IBCNEDE7(2) ; Get setting for pre reg. extract I 'SETSTR Q ; Quit if extract is not active S SLCCRIT1=\$P(SETSTR,U,2) ; Selection Criteria #1 S MAXCNT=\$P(SETSTR,U,4) ; Max # of TQ entries to create S:MAXCNT="" MAXCNT=9999999999 S SUPPBUFF=\$P(SETSTR,U,5) ; Suppress Buffer Flag S FRESHDAY=\$P(\$G(^IBE(350.9,1,51)),U,1) ; Freshness days span S CNT=0 ; Init. TQ entry counter S ENDDT=\$\$FMADD^XLFD(DT,SLCCRIT1) ; End of appt. date selection range S IBCNETOT=0 ; Initialize count for periodic TaskMan check K ^TMP(\$J,"SDAMA301"),^TMP("IBCNEDE2",\$J) ; Clean TMP globals ; S CLNC=0 ; Init. clinic ; Loop through clinics F S CLNC=\$O(^SC(CLNC)) Q:'CLNC!(CNT<MAXCNT) D Q:\$G(ZTSTOP) . ; . D CLINICEX Q:'OK ; Check for clinic exclusion . ; . S ^TMP("IBCNEDE2",\$J,CLNC)="" ; ; Set up variables for scheduling call and call S IBSDA("FLDS")=8 S IBSDA(1)=DT_"", "_ENDDT S IBSDA(2)="^TMP("IBCNEDE2",\$J," S IBSDA(3)="R" </pre>	

Routine Name	EN^IBCNEDE2
	<pre> S NUM=\$\$SDAPI^SDAMA301(.IBSDA) I NUM<1 D:NUM<0 ERRMSG G ENQ ; ; ; S CLNC=0 ; Init. clinic ; Loop through clinics returned F S CLNC=\$O(^TMP(\$J,"SDAMA301",CLNC)) Q:'CLNC D Q:\$G(ZTSTOP)!(CNT'<MAXCNT) . ; . ; Loop through patients returned . S DFN=0 F S DFN=\$O(^TMP(\$J,"SDAMA301",CLNC,DFN)) Q:'DFN!(CNT'<MAXCNT) D Q:\$G(ZTSTOP) .. ; .. S APTDT=DT ; Check for appointment date .. S MCAREFLG=0 .. ; .. ; Loop through dates in range at clinic .. F S APTDT=\$O(^TMP(\$J,"SDAMA301",CLNC,DFN,APTDT)) Q:('APTDT)!((APTDT\1)>ENDDT)!(CNT'<MAXCNT) D Q:\$G(ZTSTOP) ... ; ... S SRVICEDT=APTDT\1 ;Set service date equal to appointment date ... S FRESHDT=\$\$FMADD^XLFD(TSRVICEDT,-FRESHDAY) ... ; ... ; Update count for periodic check ... S IBCNETOT=IBCNETOT+1 ... ; Check for request to stop background job, periodically ... I \$D(ZTQUEUEU),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q ... ; ... S IBSDATA=\$G(^TMP(\$J,"SDAMA301",CLNC,DFN,APTDT)) ... S ELG=\$P(IBSDATA,U,8) ... S ELG=\$\$S(ELG'="":ELG,1:\$P(\$G(^DPT(DFN,.36)),U,1)) ... I \$P(\$G(^DPT(DFN,0)),U,21) Q ; Exclude if test patient ... I \$P(\$G(^DPT(DFN,.35)),^",1)'="" Q ; Exclude if patient is deceased ... ; ... D ELG Q:'OK ; Check for eligibility exclusion ... ; ... K ACTINS ... D ALL^IBCNS1(DFN,"ACTINS",2) ... ; ... I \$D(ACTINS(0)) Q ; Patient has no active ins ... ; ... S INREC=0 ; Record ien ... F S INREC=\$O(ACTINS(INREC)) Q:('INREC)!(CNT'<MAXCNT) D S INSIEN=\$P(\$G(ACTINS(INREC,0)),U,1) ; Insurance ien S INSNAME=\$P(\$G(^DIC(36,INSIEN,0)),U) ; exclude policies that have been verified within "freshness days" S VDATE=\$P(\$G(ACTINS(INREC,1)),U,3) I VDATE'="",SRVICEDT'>\$\$FMADD^XLFD(TVDATE,FRESHDAY) Q ; allow only one MEDICARE transmission per patient I INSNAME["MEDICARE",MCAREFLG Q ; exclude pharmacy policies I \$\$GET1^DIQ(36,INSIEN_",",.13)="PRESCRIPTION ONLY" Q S GIEN=+\$P(\$G(ACTINS(INREC,0)),U,18) I GIEN,\$\$GET1^DIQ(355.3,GIEN_",",.09)="PRESCRIPTION" Q ; check for ins. to exclude (i.e. Medicaid) </pre>

Routine Name	EN^IBCNEDE2
	<pre> I \$\$EXCLUDE^IBCNEUT4(INSNAME) Q ; check insurance policy expiration date I \$\$EXPIRED(\$P(\$G(ACTINS(INREC,0)),U,4)) Q ; ; set patient id field IB*2*416 S PATID=\$P(\$G(ACTINS(INREC,5)),U,1) ; 5.01 field ; S PAYERSTR=\$\$INSERROR^IBCNEUT3("I",INSIEN) ; Get payer info ; S SYMBOL=+PAYERSTR ; error symbol S PAYERID=\$P(PAYERSTR,U,3) ; (National ID) payer id S PIEN=\$P(PAYERSTR,U,2) ; Payer ien I '\$\$PYRACTV^IBCNEDE7(PIEN) Q ; Payer is not nationally active ; ; If error symbol exists, set record in insurance buffer & quit I SYMBOL D Q I 'SUPPBUFF,\$\$BFEXIST^IBCNEUT5(DFN,INSNAME) D PT^IBCNEBF(DFN,INREC,SYMBOL,"",1) ; ; Update service date and freshness date based on payers allowed ; date range D UPDDTS^IBCNEDE6(PIEN,.SRVICEDT,.FRESHDT) ; ; Update service dates for inquiry to be transmitted D TQUPDSV^IBCNEUT5(DFN,PIEN,SRVICEDT) ; ; Quit before filing if outstanding entries in TQ I '\$\$ADDTQ^IBCNEUT5(DFN,PIEN,SRVICEDT,FRESHDAY) Q ; S QUERYFLAG="V" K SIDARRAY S SIDDATA=\$\$SIDCHK^IBCNEDE5(PIEN,DFN,..SIDARRAY,FRESHDT) S SIDACT=\$P(SIDDATA,U),SIDCNT=\$P(SIDDATA,U,2) I SIDACT=3,'SUPPBUFF,\$\$BFEXIST^IBCNEUT5(DFN,INSNAME) D PT^IBCNEBF(DFN,INREC,18,"",1) Q I CNT+SIDCNT>MAXCNT S CNT=MAXCNT Q ;exceeds MAXCNT ; S SID="" F S SID=\$O(SIDARRAY(SID)) Q:SID="" D:\$P(SID,"_")="" SET(\$P(SID,"_"),\$P(SID,"_",2),PATID) S:INSNAME["MEDICARE" MCAREFLG=1 I SIDACT=4 D SET("", "",PATID) S:INSNAME["MEDICARE" MCAREFLG=1 Q ... Q ENQ K ^TMP(\$J,"SDAMA301"),^TMP("IBCNEDE2",\$J) Q ; </pre>
Modified Logic (Changes are in bold)	
	<p>Modified to create an Insurance Buffer record with a blank symbol for instances where the payer is "nationally inactive":</p> <pre> ; EN ; Loop through designated cross-references for updates </pre>

Routine Name	EN^IBCNEDE2
	<pre> ; Pre reg extract (Appointment extract) ; N TODAYSDT,FRESHDAY,SLCCRIT1,MAXCNT,CNT,ENDDT,CLNC,FRESHDT,GIEN N APTDT,INREC,INSIEN,PAYER,PIEN,PAYERSTR,SYMBOL,SUPPBUFF,PATID N DFN,OK,VAIN,INS,DATA1,DATA2,ELG,PAYERID,SETSTR,SRVICEDT,ACTINS N TQIEN,IBINDT,IBOUTP,QUERYFLAG,INSNAME,FOUND1,FOUND2,IBCNETOT,VDATE N SID,SIDACT,SIDDATA,SIDARRAY,SIDCNT,IBDDI,IBINS,DISYS,NUM,MCAREFLG ; S SETSTR=\$\$SETTINGS^IBCNEDE7(2) ; Get setting for pre reg. extract I 'SETSTR Q ; Quit if extract is not active S SLCCRIT1=\$P(SETSTR,U,2) ; Selection Criteria #1 S MAXCNT=\$P(SETSTR,U,4) ; Max # of TQ entries to create S:MAXCNT="" MAXCNT=9999999999 S SUPPBUFF=\$P(SETSTR,U,5) ; Suppress Buffer Flag S FRESHDAY=\$P(\$G(^IBE(350.9,1,51)),U,1) ; Freshness days span S CNT=0 ; Init. TQ entry counter S ENDDT=\$\$FMADD^XLFD(T,SLCCRIT1) ; End of appt. date selection range S IBCNETOT=0 ; Initialize count for periodic TaskMan check K ^TMP(\$J,"SDAMA301"),^TMP("IBCNEDE2",\$J) ; Clean TMP globals ; S CLNC=0 ; Init. clinic ; Loop through clinics F S CLNC=\$O(^SC(CLNC)) Q:'CLNC!(CNT<MAXCNT) D Q:\$G(ZTSTOP) . ; . D CLINICEX Q:'OK ; Check for clinic exclusion . ; . S ^TMP("IBCNEDE2",\$J,CLNC)="" ; ; Set up variables for scheduling call and call S IBSDA("FLDS")=8 S IBSDA(1)=DT_"",_ENDDT S IBSDA(2)="^TMP("IBCNEDE2",\$J," S IBSDA(3)="R" S NUM=\$\$SDAPI^SDAMA301(.IBSDA) I NUM<1 D:NUM<0 ERRMSG G ENQ ; ; S CLNC=0 ; Init. clinic ; Loop through clinics returned F S CLNC=\$O(^TMP(\$J,"SDAMA301",CLNC)) Q:'CLNC D Q:\$G(ZTSTOP)!(CNT<MAXCNT) . ; . ; Loop through patients returned . S DFN=0 F S DFN=\$O(^TMP(\$J,"SDAMA301",CLNC,DFN)) Q:'DFN!(CNT<MAXCNT) D Q:\$G(ZTSTOP) . ; . S APTDT=DT ; Check for appointment date . S MCAREFLG=0 . ; . ; Loop through dates in range at clinic . F S APTDT=\$O(^TMP(\$J,"SDAMA301",CLNC,DFN,APTDT)) Q:('APTDT)!((APTDT\1)>ENDDT)!(CNT<MAXCNT) D Q:\$G(ZTSTOP) . ; . S SRVICEDT=APTDT\1 ;Set service date equal to appointment date . S FRESHDT=\$\$FMADD^XLFD(SRVICEDT,-FRESHDAY) </pre>

Routine Name	EN^IBCNEDE2
<pre> ... ; ... ; Update count for periodic check ... S IBCNETOT=IBCNETOT+1 ... ; Check for request to stop background job, periodically ... I \$D(ZTQUEUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q ... ; ... S IBSDATA=\$G(^TMP(\$J,"SDAMA301",CLNC,DFN,APTD)) ... S ELG=\$P(IBSDATA,U,8) ... S ELG=\$S(ELG=""':ELG,1:\$P(\$G(^DPT(DFN,.36)),U,1)) ... I \$P(\$G(^DPT(DFN,0)),U,21) Q ; Exclude if test patient ... I \$P(\$G(^DPT(DFN,.35)),^",1)'="" Q ; Exclude if patient is deceased ... ; ... D ELG Q:'OK ; Check for eligibility exclusion ... ; ... K ACTINS ... D ALL^IBCNS1(DFN,"ACTINS",2) ... ; ... I ' \$D(ACTINS(0)) Q ; Patient has no active ins ... ; ... S INREC=0 ; Record ien ... F S INREC=\$O(ACTINS(INREC)) Q:(INREC)!(CNT'<MAXCNT) D S INSIEN=\$P(\$G(ACTINS(INREC,0)),U,1) ; Insurance ien S INSNAME=\$P(\$G(^DIC(36,INSIEN,0)),U) ; exclude policies that have been verified within "freshness days" S VDATE=\$P(\$G(ACTINS(INREC,1)),U,3) I VDATE=""',SRVICEDT'>\$FMADD^XLFD(TVDATE,FRESHDAY) Q ; allow only one MEDICARE transmission per patient I INSNAME["MEDICARE",MCAREFLG Q ; exclude pharmacy policies I \$\$GET1^DIQ(36,INSIEN_",",.13)="PRESCRIPTION ONLY" Q S GIEN=+\$P(\$G(ACTINS(INREC,0)),U,18) I GIEN,\$\$GET1^DIQ(355.3,GIEN_",",.09)="PRESCRIPTION" Q ; check for ins. to exclude (i.e. Medicaid) I \$\$EXCLUDE^IBCNEUT4(INSNAME) Q ; check insurance policy expiration date I \$\$EXPIRED(\$P(\$G(ACTINS(INREC,0)),U,4)) Q ... ; ; set patient id field IB*2*416 S PATID=\$P(\$G(ACTINS(INREC,5)),U,1) ; 5.01 field ... ; S PAYERSTR=\$\$INSERROR^IBCNEUT3("I",INSIEN) ; Get payer info ... ; S SYMBOL=+PAYERSTR ; error symbol S PAYERID=\$P(PAYERSTR,U,3) ; (National ID) payer id S PIEN=\$P(PAYERSTR,U,2) ; Payer ien ; I '\$\$PYRACTV^IBCNEDE7(PIEN) Q ; Payer is not nationally active ... ; ; If Payer is Nationally Inactive create an Insurance Buffer record w/blank SYMBOL & quit. I '\$\$PYRACTV^IBCNEDE7(PIEN) D Q S SYMBOL="" I 'SUPPBUFF,\$\$BFEXIST^IBCNEUT5(DFN,INSNAME) D PT^IBCNEBF(DFN,INREC,SYMBOL,"",1) ... ; </pre>	

Routine Name	EN^IBCNEDE2
<pre> ; If error symbol exists, set record in insurance buffer & quit I SYMBOL D Q I 'SUPPBUFF,'\$\$BFEXIST^IBCNEUT5(DFN,INSNAME) D PT^IBCNEBF(DFN,INREC,SYMBOL,"",1) ; ; Update service date and freshness date based on payers allowed ; date range D UPDDTS^IBCNEDE6(PIEN,.SRVICEDT,.FRESHDT) ; ; Update service dates for inquiry to be transmitted D TQUPDSV^IBCNEUT5(DFN,PIEN,SRVICEDT) ; ; Quit before filing if outstanding entries in TQ I '\$\$ADDTQ^IBCNEUT5(DFN,PIEN,SRVICEDT,FRESHDAY) Q ; S QUERYFLAG="V" K SIDARRAY S SIDDATA=\$\$SIDCHK^IBCNEDE5(PIEN,DFN,.,SIDARRAY,FRESHDT) S SIDACT=\$P(SIDDATA,U),SIDCNT=\$P(SIDDATA,U,2) I SIDACT=3,'SUPPBUFF,'\$\$BFEXIST^IBCNEUT5(DFN,INSNAME) D PT^IBCNEBF(DFN,INREC,18,"",1) Q I CNT+SIDCNT>MAXCNT S CNT=MAXCNT Q ;exceeds MAXCNT ; S SID="" F S SID=\$O(SIDARRAY(SID)) Q:SID="" D:\$P(SID,"_")="" SET(\$P(SID,"_"),\$P(SID,"_",2),PATID) S:INSNAME["MEDICARE" MCAREFLG=1 I SIDACT=4 D SET(""," ",PATID) S:INSNAME["MEDICARE" MCAREFLG=1 Q ... Q ENQ K ^TMP(\$J,"SDAMA301"),^TMP("IBCNEDE2",\$J) Q ; </pre>	

Routine Name	IBCNBLA1	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.2.20 , 2.6.1.21 & 2.6.1.22	
Related Options	“Escalate” action for Insurance Buffer Lists	
Related Routines	Routines “Called By”	Routines “Called”
	New protocol: “IBCNB ENTRY ESCALATE” Described below in another chart.	DISPBUF^IBCNBU1 EDITSTF^IBCNBES INIT^IBCNBLE HDR^IBCNBLE UPDLN^IBCNBLL
Data Dictionary (DD)		

Routine Name	IBCNBLA1
References	
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
No tag exists for the new action "Escalate"	
Modified Logic (Changes are in bold)	
<pre> ; ESCALATE(IBBUFDA) ; escalate a buffer entry ; N DIR,DIRUT,X,Y,IBX,IBY Q: '\$G(IBBUFDA) I '\$D(^XUSEC("IB INSURANCE COMPANY EDIT",DUZ))&(' \$D(^XUSEC("IB GROUP/PLAN EDIT",DUZ)) D G ESCX . W "Sorry you are not authorized to ESCALATE an entry." H 2 D FULL^VALM1 S VALMBCK="R" W ! D DISPBUF^IBCNBU1(IBBUFDA) ; S IBX=\$G(^IBA(355.33,IBBUFDA,0)),IBY=" " ; S DIR("?")="Enter Yes if the Buffer entry is not an existing active policy on the patient's file or if you can't process this entry." W !! S DIR(0)="YO",DIR("B")="N",DIR("A")=IBY_"Escalate this buffer entry" D ^DIR I Y=1 D . K IBX S IBX(.1)="NOW",IBX(.12)="E1" D EDITSTF^IBCNBES(IBBUFDA,.IBX) . D INIT^IBCNBLE,HDR^IBCNBLE S VALMBCK="R" D UPDLN^IBCNBLL(IBBUFDA,"EDITED") W " Entry Escalated ..." H 2 ESCX ; Q </pre>	

3.2.3 Data Dictionaries

File Name and Number	INSURANCE BUFFER (#355.33)
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Requirements Traceability Matrix	2.6.1.18
Related Options	Process Insurance Buffer [IBCN INSURANCE BUFFER PROCESS]
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
File Documentation	
File Auditing, Security, and Archiving	

Field Name	NAME OF FILE
Field Description	INSURANCE VERIFICATION PROCESSOR
Field #	
Node #	^IBA(355.33,0)
Piece #	1
New Field	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Data Type	<input type="checkbox"/> Date/Time <input type="checkbox"/> Numeric <input type="checkbox"/> Set of Codes <input checked="" type="checkbox"/> Free Text
	<input type="checkbox"/> Pointer to a File <input type="checkbox"/> Variable-Pointer
Identifier	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Uneditable Field	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Mandatory Field	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Field Documentation or Help Changes Necessary	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Field Definition	Name of File
Input/Output Transform	
Cross-Reference (id and type)	<input type="checkbox"/> Regular <input type="checkbox"/> Kwic <input type="checkbox"/> Mnemonic <input type="checkbox"/> Mumps
No cross reference	<input type="checkbox"/> Soundex <input type="checkbox"/> Trigger <input type="checkbox"/> Bulletin

3.2.4 Protocols

Protocol Name	IBCNB LIST COMPLETE VIEW
Enhancement Category	<input checked="" type="checkbox"/> New <input type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Associated Protocols	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output <input type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference
Item Text Description	Complete View
Protocol Type	<input checked="" type="checkbox"/> Action <input type="checkbox"/> Menu <input type="checkbox"/> Protocol <input type="checkbox"/> Protocol Menu <input type="checkbox"/> Limited Protocol <input type="checkbox"/> Extended Action <input type="checkbox"/> Dialog <input type="checkbox"/> Other
Associated Routine	IBCNBL*
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
K IBCNSORT D EN1^ IBCNBLL (6)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	IBCNB LIST FAILURE VIEW
Enhancement Category	<input checked="" type="checkbox"/> New <input type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Associated Protocols	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output <input type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference
Item Text Description	Complete View
Protocol Type	<input checked="" type="checkbox"/> Action <input type="checkbox"/> Menu <input type="checkbox"/> Protocol <input type="checkbox"/> Protocol Menu <input type="checkbox"/> Limited Protocol <input type="checkbox"/> Extended Action <input type="checkbox"/> Dialog <input type="checkbox"/> Other
Associated Routine	IBCNBL*
Current Entry Action Logic	

Modified Entry Action Logic (Changes are in bold)
K IBCNSORT D EN1^IBCNBLL (7)
Current Exit Action Logic
Modified Exit Action Logic (Changes are in bold)

Protocol Name	IBCNB ENTRY ESCALATE
Enhancement Category	<input checked="" type="checkbox"/> New <input type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Associated Protocols	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output <input type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference
Item Text Description	Escalate
Protocol Type	<input checked="" type="checkbox"/> Action <input type="checkbox"/> Menu <input type="checkbox"/> Protocol <input type="checkbox"/> Protocol Menu <input type="checkbox"/> Limited Protocol <input type="checkbox"/> Extended Action <input type="checkbox"/> Dialog <input type="checkbox"/> Other
Associated Routine	IBCNBL*
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
D ESCALATE^IBCNBLA1 (IBBUFDA)	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	IBCNB LIST APPOINTMENTS VIEW
Enhancement Category	<input type="checkbox"/> New <input type="checkbox"/> Modify <input checked="" type="checkbox"/> Delete <input type="checkbox"/> No Change
Associated Protocols	
Data Passing	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> Both <input type="checkbox"/> Global <input type="checkbox"/> Local Reference

	Input	Output	Reference
Item Text Description	N/A		
Protocol Type	<input type="checkbox"/> Action <input type="checkbox"/> Limited Protocol <input type="checkbox"/> Other	<input type="checkbox"/> Menu <input type="checkbox"/> Extended Action	<input type="checkbox"/> Protocol <input type="checkbox"/> Protocol Menu <input type="checkbox"/> Dialog
Associated Routine	IBCNBL*		
Current Entry Action Logic			
Modified Entry Action Logic (Changes are in bold)			
Current Exit Action Logic			
Modified Exit Action Logic (Changes are in bold)			

Protocol Name	IBCNB ENTRY VERIFY			
Enhancement Category	<input type="checkbox"/> New <input type="checkbox"/> Modify <input checked="" type="checkbox"/> Delete <input type="checkbox"/> No Change			
Associated Protocols				
Data Passing	<input type="checkbox"/> Input	<input type="checkbox"/> Output	<input type="checkbox"/> Both	<input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference
Item Text Description	N/A			
Protocol Type	<input type="checkbox"/> Action <input type="checkbox"/> Limited Protocol <input type="checkbox"/> Other	<input type="checkbox"/> Menu <input type="checkbox"/> Extended Action	<input type="checkbox"/> Protocol <input type="checkbox"/> Dialog	<input type="checkbox"/> Protocol Menu
Associated Routine				
Current Entry Action Logic				
Modified Entry Action Logic (Changes are in bold)				
Current Exit Action Logic				

Modified Exit Action Logic (Changes are in bold)

Protocol Name	IBCNB LIST SCREEN MENU
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Associated Protocols	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output <input type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference
Item Text Description	N/A
Protocol Type	<input type="checkbox"/> Action <input type="checkbox"/> Menu <input type="checkbox"/> Protocol <input checked="" type="checkbox"/> Protocol Menu <input type="checkbox"/> Limited Protocol <input type="checkbox"/> Extended Action <input type="checkbox"/> Dialog <input type="checkbox"/> Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Item Logic	
ITEM: IBCNB LIST PROCESS SCREEN MNEMONIC: PE SEQUENCE: 11 ITEM: IBCNB LIST REJECT MNEMONIC: RE SEQUENCE: 12 ITEM: IBCNB LIST ENTRY SCREEN MNEMONIC: EE SEQUENCE: 13 ITEM: IBCNB LIST ADD MNEMONIC: AE SEQUENCE: 21 ITEM: IBCNB LIST SORT MNEMONIC: ST SEQUENCE: 22 ITEM: IBCNB FAST EXIT MNEMONIC: EX SEQUENCE: 43 ITEM: IBCNB LIST CHECK NAMES MNEMONIC: CC SEQUENCE: 23 ITEM: IBCNB LIST POSITIVE VIEW MNEMONIC: PB SEQUENCE: 31 ITEM: IBCNB LIST NEGATIVE VIEW MNEMONIC: NB SEQUENCE: 32 ITEM: IBCNB LIST MEDICARE VIEW MNEMONIC: MB SEQUENCE: 33 ITEM: IBCNB LIST APPOINTMENTS VIEW MNEMONIC: FA SEQUENCE: 41 ITEM: IBCNB LIST EPHARMACY VIEW MNEMONIC: RX SEQUENCE: 42	
Modified ItemLogic (Changes are in bold)	
ITEM: IBCNB LIST PROCESS SCREEN MNEMONIC: PE	

SEQUENCE: 11	
ITEM: IBCNB LIST REJECT	MNEMONIC: RE
SEQUENCE: 12	
ITEM: IBCNB LIST ENTRY SCREEN	MNEMONIC: EE
SEQUENCE: 13	
ITEM: IBCNB LIST ADD	MNEMONIC: AE
SEQUENCE: 21	
ITEM: IBCNB LIST SORT	MNEMONIC: ST
SEQUENCE: 22	
ITEM: IBCNB FAST EXIT	MNEMONIC: EX
SEQUENCE: 43	
ITEM: IBCNB LIST CHECK NAMES	MNEMONIC: CC
SEQUENCE: 23	
ITEM: IBCNB LIST POSITIVE VIEW	MNEMONIC: PB
SEQUENCE: 31	
ITEM: IBCNB LIST NEGATIVE VIEW	MNEMONIC: NB
SEQUENCE: 32	
ITEM: IBCNB LIST MEDICARE VIEW	MNEMONIC: MB
SEQUENCE: 33	
ITEM: IBCNB FAILURE VIEW	MNEMONIC: FB
SEQUENCE: 41	
ITEM: IBCNB LIST EPHARMACY VIEW	MNEMONIC: RX
SEQUENCE: 42	
ITEM: IBCNB COMPLETE VIEW	MNEMONIC: CB
SEQUENCE: 44	
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

Protocol Name	IBCNB ENTRY SCREEN MENU
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Associated Protocols	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output <input type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference
Item Text Description	N/A
Protocol Type	<input type="checkbox"/> Action <input type="checkbox"/> Menu <input type="checkbox"/> Protocol <input checked="" type="checkbox"/> Protocol Menu <input type="checkbox"/> Limited Protocol <input type="checkbox"/> Extended Action <input type="checkbox"/> Dialog <input type="checkbox"/> Other
Associated Routine	
Current Entry Action Logic	
Modified Entry Action Logic (Changes are in bold)	
Current Item Logic	
ITEM: IBCNB ENTRY EDIT INSURANCE	MNEMONIC: EI
SEQUENCE: 11	

ITEM: IBCNB ENTRY EDIT ALL SEQUENCE: 12	MNEMONIC: EA
ITEM: IBCNB ENTRY EDIT GROUP SEQUENCE: 13	MNEMONIC: PE
ITEM: IBCNB ENTRY VERIFY SEQUENCE: 21	MNEMONIC: VE
ITEM: IBCNB ENTRY EDIT POLICY SEQUENCE: 22	MNEMONIC: PI
ITEM: IBCNB FAST EXIT SEQUENCE: 32	MNEMONIC: EX
ITEM: IBCNB ENTRY RESPONSE REPORT SEQUENCE: 23	MNEMONIC: RR
ITEM: IBCNB EXPAND BENEFITS SEQUENCE: 31	MNEMONIC: EB
Modified ItemLogic (Changes are in bold)	
ITEM: IBCNB ENTRY EDIT INSURANCE SEQUENCE: 11	MNEMONIC: EI
ITEM: IBCNB ENTRY EDIT ALL SEQUENCE: 12	MNEMONIC: EA
ITEM: IBCNB ENTRY EDIT GROUP SEQUENCE: 13	MNEMONIC: PE
ITEM: IBCNB ENTRY ESCALATE SEQUENCE: 21	MNEMONIC: ES
ITEM: IBCNB ENTRY EDIT POLICY SEQUENCE: 22	MNEMONIC: PI
ITEM: IBCNB FAST EXIT SEQUENCE: 32	MNEMONIC: EX
ITEM: IBCNB ENTRY RESPONSE REPORT SEQUENCE: 23	MNEMONIC: RR
ITEM: IBCNB EXPAND BENEFITS SEQUENCE: 31	MNEMONIC: EB
Current Exit Action Logic	
Modified Exit Action Logic (Changes are in bold)	

3.3 System Feature: eIV HL7 Transactions

This section covers the design for the requirements listed in Section 2.6.2 of the RSD.

3.3.1 Functional Requirements:

Table 2: eIV HL7 Transactions

SRS REQ ID	REQ Title	Comments / Notes
2.6.2.1	eIV HL7 Transactions - Daily Registration Message to FSC	
2.6.2.2	eIV HL7 Transactions – Receive Retry flag from FSC	
2.6.2.3	eIV HL7 Transactions – Store Retry flag from FSC	
2.6.2.4	eIV HL7 Transactions – Receive Freshness Days from FSC	

SRSRSD REQ ID	REQ Title	Comments / Notes
2.6.2.5	eIV HL7 Transactions – Store Freshness Days from FSC	
2.6.2.6	eIV HL7 Transactions – Receive Timeout Days from FSC	
2.6.2.7	eIV HL7 Transactions – Store Timeout Days from FSC	
2.6.2.8	eIV HL7 Transactions – Treat all AAA Action Codes as Though the Payer/FSC Responded	Need to do a data conversion in a post install routine in addition to changing ^IBCNEHL3. The post install code is addressed in section 3.4.2 of this SDD. (Routine IBY506PO)
2.6.2.9	eIV HL7 Transactions – Honor the Retry Flag when Resending an eIV Inquiry	
2.6.2.10	eIV HL7 Transactions – Honor the Timeout Days when Resending an eIV Inquiry	
2.6.2.11	eIV HL7 Transactions – Honor the '# of Retries' when Resending an eIV Inquiry	
2.6.2.12	eIV HL7 Transactions – Honor the Payer's Nationally Active Flag when Resending an eIV Inquiry	
2.6.2.13	eIV HL7 Transactions - Do Not Send MailMan Message When Retries are Exhausted	

3.3.2 Routines

Routine Name	IBCNEHLM	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.2.1	
Related Options	HL7 Registration MFN Message	
Related Routines	Routines “Called By”	Routines “Called”
	^IBCNEDE	MGRP^IBCNEUT5 INIT^HLFNC2 SITE^VASITE HLP^IBCNEHLU MFE^VAFHLMFE MFI^VAFHLMFI HLNAME^HLFNC GENERATE^HLMA
Data Dictionary (DD) References		

Routine Name	IBCNEHLM
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
<p>The current IBCNEHLM routine looks like the following:</p> <pre> IBCNEHLM ;DAOU/ALA - HL7 Registration MFN Message ;10-JUN-2002 ;;2.0;INTEGRATED BILLING;**184,251,300,416,438,497**;21-MAR-94;Build 52 ;;Per VHA Directive 2004-038, this routine should not be modified. ; ; ; **Program Description** ; This program will process the outgoing registration MFN message ; ; ; Variables ; MCT = Lines of MailMan message counter ; QFL = Quit flag ; HL* = HL7 package specific variables ; TAXID = Tax ID ; CNTCPH = Contact Phone ; CNTCEM = Contact Email ; FRSH = Freshness Days ; MGRP = Mailgroup to email messages to ; INACT = Inactive Insurance Flag ; CNTC = Contact IEN ; APP = Application ; EVENT = HL7 Event ; CODE = Values sent in the MFN message ; IPP = IP Port ; IPA = IP Address ; RESP = Response Code ; IHLP = Interface HL7 Processing Type ; IHLT = Interface HL7 Batch Start Time ; IHLS = Interface HL7 Batch Stop Time ; IVER = Interface Version ; ; REG ; Registration message for when a site installs NEW TAXID,CNTCPH,CNTCEM,CNTCNM,IBCNE,FRSH,MGRP,INACT,IHLP,MFE,HLSAN NEW IHLT,CNTC,APP,EVENT,CODE,EDT,MFN,HL,HLFS,HLECH,MCT,HLPROD,HLX,ID NEW HLEID,IPP,IPA,IBCNEDAT,HLCS,HLINST,HLN,RESP,HLHDR,HLREP NEW HLTYPE,HLQ,HLRESLT,IHLS,HLCDOM,HLCINS,HLCSTCP,HLIP,%I,ZMID NEW VMFE,IVER K ^TMP("HLS",,\$J) S MCT=0,QFL=0 </pre>	

Routine Name	IBCNEHLM
	<pre> ; ; ; Get data from IB Parameters File S TAXID=\$TR(\$P(\$G(^IBE(350.9,1,1)),U,5),"-",""),CNTCPH="",CNTCEM="",CNTCNM="" S IBCNE=\$G(^IBE(350.9,1,51)) S FRSH=\$P(IBCNE,U,1) S MGRP=\$MGRP^IBCNEUT5() S INACT=\$E(\$\$GET1^DIQ(350.9,"1","51.08,"E")) S IHLP=\$P(IBCNE,U,13),IHLT=\$P(IBCNE,U,14),CNTC=\$P(IBCNE,U,16) S IHLS=\$P(IBCNE,U,19) S IVER="6" ; I IHLP="I" S (IHLT,IHLS)=" " ; ; Get contact specific information I CNTC="" D . S CNTCNM=\$P(\$G(^VA(200,CNTC,0)),U,1) . S CNTCPH=\$P(\$G(^VA(200,CNTC,.13)),U,2) . S CNTCEM=\$P(\$G(^VA(200,CNTC,.15)),U,1) ; ; Email if any missing data I CNTC="" S MCT=MCT+1,MSG(MCT)="The Contact Person is not defined in the eIV Site Parameters. ",QFL=1 I CNTC="",CNTCPH="" S MCT=MCT+1,MSG(MCT)="The office phone number of the eIV Contact Person is not defined (File 200, Field .132). ",QFL=1 I CNTC="",CNTCEM="" S MCT=MCT+1,MSG(MCT)="The email address of the eIV Contact Person is not defined (File 200, Field .151). ",QFL=1 ; I IHLP="B",IHLT=""!(IHLS="") D S QFL=1 . S MCT=MCT+1,MSG(MCT)="The ""HL7 Response Processing Method"" selected is Batch but the HL7 Batch " . I IHLT="",IHLS="" S MSG(MCT)=MSG(MCT)_ "Start and End Times are blank. " Q . S MSG(MCT)=MSG(MCT)_ \$S(IHLT="": "Start",1: "End")_ " Time is blank. " ; I FRSH=""!(INACT="")!(IHLP="") D . S MCT=MCT+1,MSG(MCT)="The following eIV Site Parameters are not defined: " . I FRSH="" S MCT=MCT+1,MSG(MCT)="""Days between electronic reverification checks"" is blank. " . I INACT="" S MCT=MCT+1,MSG(MCT)="""Look at a patient's inactive insurance?"" is blank. " . I IHLP="" S MCT=MCT+1,MSG(MCT)="""HL7 Response Processing Method"" is blank. " . Q ; I \$O(MSG(""))="" D MLMN I QFL=1 Q ; HL ; When a site installs, the enrollment should be an ; "MUP" (update) record. N DSTAT,VNTE,VZRR S MFE(1)="MUP" ; ; Initialize the HL7 D INIT^HLFNC2("IBCNE IIV REGISTER",.HL) S HLFS=HL("FS"),HLECH=HL("ECH"),HL("SAF")=\$P(\$\$SITE^VASITE,U,2,3),HLREP=\$E(HL("ECH"),2) ; S HLEID=\$\$HLP^IBCNEHLU("IBCNE IIV REGISTER") ; </pre>

Routine Name	IBCNEHLM
	<pre> ; Set the MFI segment S ID="Facility Table",APP="",EVENT="UPD",RESP="NE" S ^TMP("HLS",,\$J,1)=\$\$MFI^VAFHLMFI(ID,APP,EVENT,,,RESP) ; ; Set the MFE segment S EVENT=MFE(1),MFN="",EDT=\$\$DT^XLFD() S CODE=\$P(\$\$SITE^VASITE,U,3)_E(HLECH) S VMFE=\$\$MFE^VAFHLMFE(EVENT,MFN,EDT,CODE) S ^TMP("HLS",,\$J,2)=VMFE_HLFS_"CE" ; ; Set the ZRR segment S VZRR="ZRR"_HLFS_"1"_HLFS_TAXID_HLFS_HLFS_\$\$HLNAME^HLFNC(CNTCNM,\$E(HLECH))_"^ C"_HLFS S VZRR=VZRR_CNTCPH_\$E(HLECH)_E(HLECH)_E(HLECH)_CNTCEM_HLFS_FRSH_HLFS_IHLP _HLFS_IHLT_\$E(HLECH)_IHLS_HLFS_INACT_HLFS_IVER S ^TMP("HLS",,\$J,3)=VZRR ; ; Set the NTE segment S DSTAT=\$\$GETSTAT^IBCNEEDST() S VNTE="NTE"_HLFS_"1"_HLFS_HLFS_\$TR(DSTAT,U,HLREP) S ^TMP("HLS",,\$J,4)=VNTE ; D GENERATE^HLMA("IBCNE IIV REGISTER","GM",1,,HLRESLT,"") I \$P(HLRESLT,U,2)]" S HLRESLT="Error - "\$P(HLRESLT,U,2,99) D Q . S MSG(1)="HL7 eIV Registration Message not created." . S MSG(2)=HLRESLT . D MLMN K ^TMP("HLS",,\$J) Q ; </pre>
Modified Logic (Changes are in bold)	
<p>Modifications to be made are in add 3 fields (the retry flag [#350.9,51.26],the freshness days [#350.9, 51.01], and the timeout days [#350.9, 51.05]) to the setting of the NTE segment:</p> <p>IBCNEHLM ;DAOU/ALA - HL7 Registration MFN Message ;10-JUN-2002 ;;2.0;INTEGRATED BILLING; **184,251,300,416,438,497** ;21-MAR-94;Build 52 ;;Per VHA Directive 2004-038, this routine should not be modified.</p> <pre> ; ; ; **Program Description** ; This program will process the outgoing registration MFN message ; ; Variables ; MCT = Lines of MailMan message counter ; QFL = Quit flag ; HL* = HL7 package specific variables ; TAXID = Tax ID ; CNTCPH = Contact Phone ; CNTCEM = Contact Email ; FRSH = Freshness Days ; MGRP = Mailgroup to email messages to </pre>	

Routine Name	IBCNEHLM
<pre> ; INACT = Inactive Insurance Flag ; CNTC = Contact IEN ; APP = Application ; EVENT = HL7 Event ; CODE = Values sent in the MFN message ; IPP = IP Port ; IPA = IP Address ; RESP = Response Code ; IHLP = Interface HL7 Processing Type ; IHLT = Interface HL7 Batch Start Time ; IHLS = Interface HL7 Batch Stop Time ; IVER = Interface Version ; TIMEOUT = Timeout Days Site Parameter ; RETRY = Retry Flag Site Parameter ; REG ; Registration message for when a site installs NEW TAXID,CNTCPH,CNTCEM,CNTCNM,IBCNE,FRSH,MGRP,INACT,IHLP,MFE,HLSAN NEW IHLT,CNTC,APP,EVENT,CODE,EDT,MFN,HL,HLFS,HLECH,MCT,HLPROD,HLX,ID NEW HLEID,IPP,IPA,IBCNE DAT,HLCS,HLINST,HLN,RESP,HLHDR,HLREP NEW HLTYPE,HLQ,HLRESLT,IHLS,HLCDOM,HLCINS,HLCSTCP,HLIP,%I,ZMID NEW VMFE,IVER,TIMOUT,RETRY K ^TMP("HLS",\$J) S MCT=0,QFL=0 ; ; Get data from IB Parameters File S TAXID=\$TR(\$P(\$G(^IBE(350.9,1,1)),U,5),"-", ""),CNTCPH="",CNTCEM="",CNTCNM="" S IBCNE=\$G(^IBE(350.9,1,51)) S FRSH=\$P(IBCNE,U,1),TIMOUT=\$P(IBCNE,U,5),RETRY=\$P(IBCNE,U,26) S MGRP=\$MGRP^IBCNEUT5() S INACT=\$E(\$\$GET1^DIQ(350.9,"1","51.08,"E")) S IHLP=\$P(IBCNE,U,13),IHLT=\$P(IBCNE,U,14),CNTC=\$P(IBCNE,U,16) S IHLS=\$P(IBCNE,U,19) S IVER="6" ; I IHLP="I" S (IHLT,IHLS)=" " ; ; Get contact specific information I CNTC="" D . S CNTCNM=\$P(\$G(^VA(200,CNTC,0)),U,1) . S CNTCPH=\$P(\$G(^VA(200,CNTC,.13)),U,2) . S CNTCEM=\$P(\$G(^VA(200,CNTC,.15)),U,1) ; ; Email if any missing data I CNTC="" S MCT=MCT+1,MSG(MCT)="The Contact Person is not defined in the eIV Site Parameters. ",QFL=1 I CNTC="",CNTCPH="" S MCT=MCT+1,MSG(MCT)="The office phone number of the eIV Contact Person is not defined (File 200, Field .132). ",QFL=1 I CNTC="",CNTCEM="" S MCT=MCT+1,MSG(MCT)="The email address of the eIV Contact Person is not defined (File 200, Field .151). ",QFL=1 ; I IHLP="B",IHLT=""!(IHLS="") D S QFL=1 . S MCT=MCT+1,MSG(MCT)="The ""HL7 Response Processing Method"" selected is Batch but the HL7 Batch " . I IHLT="",IHLS="" S MSG(MCT)=MSG(MCT)_"Start and End Times are blank. " Q </pre>	

Routine Name	IBCNEHLM
<pre> . S MSG(MCT)=MSG(MCT)_\$(IHLT="":"Start",1:"End")_ " Time is blank. " ; ; I FRSH=""!(INACT="")!(IHLP="") D . S MCT=MCT+1,MSG(MCT)="The following eIV Site Parameters are not defined: " . I FRSH="" S MCT=MCT+1,MSG(MCT)=""Days between electronic reverification checks"" is blank. " . I INACT="" S MCT=MCT+1,MSG(MCT)=""Look at a patient's inactive insurance?"" is blank. " . I IHLP="" S MCT=MCT+1,MSG(MCT)=""HL7 Response Processing Method"" is blank. " . Q ; ; I \$O(MSG(""))'="" D MLMN I QFL=1 Q ; ; HL ; When a site installs, the enrollment should be an ; "MUP" (update) record. N DSTAT,VNTE,VZRR S MFE(1)="MUP" ; ; Initialize the HL7 D INIT^HLFNC2("IBCNE IIV REGISTER",.HL) S HLFS=HL("FS"),HLECH=HL("ECH"),HL("SAF")=\$P(\$\$SITE^VASITE,U,2,3),HLREP=\$E(HL("ECH"),2) ; S HLEID=\$\$HLP^IBCNEHLU("IBCNE IIV REGISTER") ; ; Set the MFI segment S ID="Facility Table",APP="",EVENT="UPD",RESP="NE" S ^TMP("HLS",\$J,1)=\$\$MFI^VAFHLMFI(ID,APP,EVENT,,,RESP) ; ; Set the MFE segment S EVENT=MFE(1),MFN="",EDT=\$\$DT^XLFD() S CODE=\$P(\$\$SITE^VASITE,U,3)_\$E(HLECH) S VMFE=\$\$MFE^VAFHLMFE(EVENT,MFN,EDT,CODE) S ^TMP("HLS",\$J,2)=VMFE_HLFS_"CE" ; ; Set the ZRR segment S VZRR="ZRR"_HLFS_"1"_HLFS_TAXID_HLFS_HLFS_\$\$HLNAME^HLFNC(CNTCNM,\$E(HLECH))_"^ C"_HLFS S VZRR=VZRR_CNTCPH_\$E(HLECH)_\$E(HLECH)_\$E(HLECH)_CNTCEM_HLFS_FRSH_HLFS_IHLP _HLFS_IHLT_\$E(HLECH)_IHLS_HLFS_INACT_HLFS_IVER S ^TMP("HLS",\$J,3)=VZRR ; ; Set the NTE segment S DSTAT=\$\$GETSTAT^IBCNEEDST() S VNTE="NTE"_HLFS_"1"_HLFS_HLFS_\$TR(DSTAT,U,HLREP) S VNTE=VNTE_HLFS_RETRY_HLFS_TIMEOUT S ^TMP("HLS",\$J,4)=VNTE ; D GENERATE^HLMA("IBCNE IIV REGISTER","GM",1,,HLRESLT,"") I \$P(HLRESLT,U,2)]"" S HLRESLT="Error - "_\$P(HLRESLT,U,2,99) D Q . S MSG(1)="HL7 eIV Registration Message not created." . S MSG(2)=HLRESLT . D MLMN K ^TMP("HLS",\$J) Q ; ; </pre>	

Routine Name	IBCNEHLM

Routine Name	EN^IBCNEHLT				
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change				
Requirement Traceability Matrix	2.6.2.2 thru 2.6.2.7				
Related Options	HL7 Processing Incoming MFN Messages				
Related Routines	Routines "Called By"	Routines "Called"			
		SPAR^IBCNEHLU MSG^IBCNEUT5 MGRP^IBCNEUT5 ^IBCNRHLLT DECHL7^IBCNEHL2			
Data Dictionary (DD) References					
Related Protocols					
Related Integration Control Registrations (ICRs)					
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local				
Input Attribute Name and Definition					
Output Attribute Name and Definition					
Current Logic					
The module of code involved is EN^IBCNEHLT: ; EN ; Entry Point NEW AIEN,APIEN,APP,D0,D,DESC,DQ,DR,FILE,FLN,HEDI,ID,IEN NEW PEDI,SEG,STAT,HCT,NEWID,TSSN,REQSUB,NAFLG,NPFLG,TRUSTED NEW IBCNACT,IBCNADT,FSVDY,PSVDY NEW BPSIEN,CMIEN,DATA,DATAAP,DATABPS,DATAACM,DATE,ERROR,FIELDNO,FILENO NEW IBSEG,MSG,BUFF NEW X12TABLE,BADFMT ; ; BADFMT is true if a site with patch 300 receives an eIV message in the previous HL7 interface structure (pre-300) ; ; Build local table of file numbers to determine if response is eIV or ePHARM F D=11:1:18 S X12TABLE("365.0"_D)="" F D=21:1:28 S X12TABLE("365.0"_D)="" ; ;					

Routine Name	EN^IBCNEHLT
<pre> ; Decide if message belongs to "E-Pharm" or "eIV" S APP="" S HCT=0,ERFLG=0 F S HCT=\$O(^TMP(\$J,"IBCNEHLI",HCT)) Q:HCT="" D SPAR^IBCNEHLU I \$G(IBSEG(1))="MFI" S FILE=\$G(IBSEG(2)),FLN=\$P(FILE,\$E(HLECH,1),1) Q I ",366.01,366.02,366.03,365.12,355.3,["_",FLN_",") S APP="E-PHARM" I FLN=365.12 D . S HCT=0,BADFMT=0 . F S HCT=\$O(^TMP(\$J,"IBCNEHLI",HCT)) Q:HCT="" D Q:(APP="IIV")!BADFMT .. D SPAR^IBCNEHLU .. I \$G(IBSEG(1))="MFE",\$P(\$G(IBSEG(5)),\$E(HLECH,1),3)'="" D Q ... S BADFMT=1,APP="" ... S MSG(1)="Log a Remedy Ticket for this issue." ... S MSG(2)="Please include in the Remedy Ticket that the eIV payer tables may be out" ... S MSG(3)="of sync with the master list and will need a new copy of the payer table" ... S MSG(4)="from Austin." ... D MSG^IBCNEUT5(\$\$MGRP^IBCNEUT5(),"eIV payer tables may be out of synch with master list",MSG()) .. I \$G(IBSEG(1))="ZPA" S APP="IIV" I \$D(X12TABLE(FLN)) S APP="IIV" ; If neither eIV or ePHARM then quit I APP="" Q ; S HCT=1,NAFLG=0,NPFLG=0,D="" F S HCT=\$O(^TMP(\$J,"IBCNEHLI",HCT)) Q:HCT="" D Q:ERFLG . D SPAR^IBCNEHLU . S SEG=\$G(IBSEG(1)) . ; . I APP="E-PHARM" D .. I SEG="MFI" D ... S FILE=\$G(IBSEG(2)) ... S FLN=\$P(FILE,\$E(HLECH,1),1) ... ; ... ; Initialize MFK Message (Application Acknowledgement) variables ... ; Master File Identifier ... S DATAMFK("MFI-1")=\$G(IBSEG(2)) ... ; ... ; File-Level Event Code ... S DATAMFK("MFI-3")=\$G(IBSEG(4)) ... ; .. I SEG="MFE" D ... I \$G(FLN)="" S ERFLG=1,MSG(1)="File Number not found in MFN message" Q ... I '\$\$VFILE^DILFD(FLN) S ERFLG=1,MSG(1)="File "_FLN_" not found in the Data Dictionary" Q ... ; ... ; Initialize MFK Message (Application Acknowledgement) variables ... ; Record-Level Event Code ... S DATAMFK("MFE-1")=\$G(IBSEG(2)) ... ; ... ; Primary Key Value ... S DATAMFK("MFE-4")=\$G(IBSEG(5)) ... ; ... ; Primary Key Value Type ... S DATAMFK("MFE-5")=\$G(IBSEG(6)) </pre>	

Routine Name	EN^IBCNEHLT
	<pre> ... ; ... ; Transfer control to e-Pharmacy ... D ^IBCNRHILT Q ... ; ... ; ... ; Transfer control on other segments .. I ",ZCM,ZP0,ZPB,ZPL,ZPT,ZRX,"[("," _SEG_",") D ^IBCNRHILT .. ; .. ; .. I APP="IIV" D .. I SEG="MFI" D ... S FILE=\$G(IBSEG(2)) ... S FLN=\$P(FILE,\$E(HLECH,1),1) .. ; .. I SEG="MFE" D ... I \$G(FLN)=" " S ERFLG=1,MSG(1)="File Number not found in MFN message" Q ... I '\$\$VFILE^DILFD(FLN) S ERFLG=1,MSG(1)="File "_FLN_" not found in the Data Dictionary" Q ... ; ... I FLN'=365.12 D Q S DATA=\$G(IBSEG(5)) S ID=\$\$DECHL7^IBCNEHL2(\$P(DATA,\$E(HLECH,1),1)),DESC=\$\$DECHL7^IBCNEHL2(\$P(DATA,\$E(H LECH,1),2)) D TFIL ... ; ... ; Pull the action code ... S IBCNACT=\$G(IBSEG(2)) ... ; Effective Date ... S IBCNADT=\$G(IBSEG(4)) ... ; .. I SEG="ZP0" D ... S ID=\$\$DECHL7^IBCNEHL2(IBSEG(3)),NEWID=\$\$DECHL7^IBCNEHL2(IBSEG(4)) ... S DESC=\$\$DECHL7^IBCNEHL2(IBSEG(5)),HEDI=\$\$DECHL7^IBCNEHL2(IBSEG(6)),PEDI=\$\$DECHL7 ^IBCNEHL2(IBSEG(7)) .. ; .. I SEG="ZPA" D ... S STAT=\$S(IBSEG(4)="Y":1,1:0) ... S TSSN=IBSEG(5),REQSUB=IBSEG(7) ... S FSVDY=IBSEG(8),PSVDY=IBSEG(9) ... S TRUSTED=\$S(IBSEG(10)="N":0,1:1) ... D PFIL Q ; </pre>
	<p>Modified Logic (Changes are in bold)</p> <p>Modifications to be made are to RECEIVE and STORE 3 fields (the freshness days [#350.9, 51.01], the timeout days [#350.9, 51.02] and the retry flag [#350.9,51.26]) and this should ONLY happen for the “eIV” application:</p> <pre> ; EN ; Entry Point NEW AIEN,APIEN,APP,D0,D,DESC,DQ,DR,FILE,FLN,HEDI,ID,IEN NEW PEDI,SEG,STAT,HCT,NEWID,TSSN,REQSUB,NAFLG,NPFLG,TRUSTED </pre>

Routine Name	EN^IBCNEHLT
<pre> NEW IBCNACT,IBCNADT,FSVDY,PSVDY NEW BPSIEN,CMIEN,DATA,DATAAP,DATABPS,DATAACM,DATE,ERROR,FIELDNO,FILENO NEW IBSEG,MSG,BUFF NEW X12TABLE,BADFMT ; ; BADFMT is true if a site with patch 300 receives an eIV message in the previous HL7 interface structure (pre-300) ; ; Build local table of file numbers to determine if response is eIV or ePHARM F D=11:1:18 S X12TABLE("365.0"_D)=" " F D=21:1:28 S X12TABLE("365.0"_D)=" " S X12TABLE("350.9")=" " ; ; Decide if message belongs to "E-Pharm" or "eIV" S APP=" " S HCT=0,ERFLG=0 F S HCT=\$O(^TMP(\$J,"IBCNEHLI",HCT)) Q:HCT=" " D SPAR^IBCNEHLU I \$G(IBSEG(1))="MFI" S FILE=\$G(IBSEG(2)),FLN=\$P(FILE,\$E(HLECH,1),1) Q I ",366.01,366.02,366.03,365.12,355.3,["(", "_FLN_", ") S APP="E-PHARM" I FLN=365.12 D . S HCT=0,BADFMT=0 . F S HCT=\$O(^TMP(\$J,"IBCNEHLI",HCT)) Q:HCT=" " D Q:(APP="IIV")!BADFMT .. D SPAR^IBCNEHLU .. I \$G(IBSEG(1))="MFE", \$P(\$G(IBSEG(5)),\$E(HLECH,1),3)'=" " D Q ... S BADFMT=1,APP=" " ... S MSG(1)="Log a Remedy Ticket for this issue." ... S MSG(2)="Please include in the Remedy Ticket that the eIV payer tables may be out" ... S MSG(3)="of sync with the master list and will need a new copy of the payer table" ... S MSG(4)="from Austin." ... D MSG^IBCNEUT5(\$MGRP^IBCNEUT5(),"eIV payer tables may be out of synch with master list",MSG(") .. I \$G(IBSEG(1))="ZPA" S APP="IIV" I \$D(X12TABLE(FLN)) S APP="IIV" ; If neither eIV or ePHARM then quit I APP=" " Q ; S HCT=1,NAFLG=0,NPFLG=0,D=" " F S HCT=\$O(^TMP(\$J,"IBCNEHLI",HCT)) Q:HCT=" " D Q:ERFLG . D SPAR^IBCNEHLU . S SEG=\$G(IBSEG(1)) . ; . I APP="E-PHARM" D .. I SEG="MFI" D ... S FILE=\$G(IBSEG(2)) ... S FLN=\$P(FILE,\$E(HLECH,1),1) ... ; ... ; Initialize MFK Message (Application Acknowledgement) variables ... ; Master File Identifier ... S DATAMFK("MFI-1")=\$G(IBSEG(2)) ... ; ... ; File-Level Event Code ... S DATAMFK("MFI-3")=\$G(IBSEG(4)) .. ; </pre>	

Routine Name	EN^IBCNEHLT
<pre> .. I SEG="MFE" D ... I \$G(FLN)=" S ERFLG=1,MSG(1)="File Number not found in MFN message" Q ... I '\$\$VFILE^DILFD(FLN) S ERFLG=1,MSG(1)="File "_FLN_" not found in the Data Dictionary" Q ... ; ... ; Initialize MFK Message (Application Acknowledgement) variables ... ; Record-Level Event Code ... S DATAMFK("MFE-1")=\$G(IBSEG(2)) ... ; ... ; Primary Key Value ... S DATAMFK("MFE-4")=\$G(IBSEG(5)) ... ; ... ; Primary Key Value Type ... S DATAMFK("MFE-5")=\$G(IBSEG(6)) ... ; ... ; Transfer control to e-Pharmacy ... D ^IBCNRHLT Q ... ; ... ; Transfer control on other segments ... I ",ZCM,ZP0,ZPB,ZPL,ZPT,ZRX,"[(","_SEG_",") D ^IBCNRHLT ... ; ... ; .. I APP="IIV" D .. I SEG="MFI" D ... S FILE=\$G(IBSEG(2)) ... S FLN=\$P(FILE,\$E(HLECH,1),1) ... ; .. I SEG="MFE" D ... I \$G(FLN)=" S ERFLG=1,MSG(1)="File Number not found in MFN message" Q ... I '\$\$VFILE^DILFD(FLN) S ERFLG=1,MSG(1)="File "_FLN_" not found in the Data Dictionary" Q ... ; ... I FLN'=365.12 D Q ... S DATA=\$G(IBSEG(5)) S ID=\$\$DECHL7^IBCNEHL2(\$P(DATA,\$E(HLECH,1),1)),DESC=\$\$DECHL7^IBCNEHL2(\$P(DATA,\$E(H LECH,1),2)) D TFIL ... ; ... ; Pull the action code ... S IBCNACT=\$G(IBSEG(2)) ... ; Effective Date ... S IBCNADT=\$G(IBSEG(4)) ... ; .. I SEG="ZP0" D ... S ID=\$\$DECHL7^IBCNEHL2(IBSEG(3)),NEWID=\$\$DECHL7^IBCNEHL2(IBSEG(4)) ... S DESC=\$\$DECHL7^IBCNEHL2(IBSEG(5)),HEDI=\$\$DECHL7^IBCNEHL2(IBSEG(6)),PEDI=\$\$DECHL7 ^IBCNEHL2(IBSEG(7)) ... ; .. I SEG="ZPA" D ... S STAT=\$S(IBSEG(4)="Y":1,1:0) ... S TSSN=IBSEG(5),REQSUB=IBSEG(7) ... S FSVDY=IBSEG(8),PSVDY=IBSEG(9) ... S TRUSTED=\$S(IBSEG(10)="N":0,1:1) ... D PFIL </pre>	

Routine Name	EN^IBCNEHLT
Q ;	

Routine Name	TFIL^IBCNEHLT			
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change			
Requirement Traceability Matrix	2.6.2.2 thru 2.6.2.7			
Related Options	HL7 Processing Incoming MFN Messages			
Related Routines	Routines "Called By"	Routines "Called"		
Data Dictionary (DD) References				
Related Protocols				
Related Integration Control Registrations (ICRs)				
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local			
Input Attribute Name and Definition				
Output Attribute Name and Definition				
Current Logic				
<p>The module of code involved is TFIL^IBCNEHLT:</p> <pre> ; TFIL ; Non Payer Tables Filer NEW DIC,X,DLAYGO,Y,IEN,MAX S DIC(0)="X",X=ID,DIC=\$\$ROOT^DILFD(FLN) D ^DIC S IEN=+Y ; don't update existing entries I IEN>0 Q D FIELD^DID(FLN,.02,,"FIELD LENGTH","MAX") I MAX("FIELD LENGTH")>0 S DESC=\$E(DESC,1,MAX("FIELD LENGTH")); restriction of the field in the DD ; add new entry to the table ;S DLAYGO=FLN,DIC(0)="L",DIC("DR")=".02///" _DESC S DLAYGO=FLN,DIC(0)="L",DIC("DR")=".02///^S X=DESC" K DD,DO D FILE^DICN K DO Q </pre>				

Routine Name	TFIL^IBCNEHLT
;	
Modified Logic (Changes are in bold)	
<p>Modifications need to be made to RECEIVE and STORE 3 fields (the freshness days [#350.9, 51.01], the timeout days [#350.9, 51.02] and the retry flag [#350.9,51.26]) and this should ONLY happen if the application is "IIV". Need to capture the FILENAME, FIELDNAME and the VALUE for these 3 fields:</p> <p>;</p> <p>TFIL ; Non Payer Tables Filer</p> <p>NEW DIC,X,DLAYGO,Y,IEN,MAX</p> <p>S DIC(0)="X",X=ID,DIC=\$\$ROOT^DILFD(FLN)</p> <p>D ^DIC S IEN=+Y</p> <p>; don't update existing entries</p> <p>I IEN>0 Q</p> <p>D FIELD^DID(FLN,.02,,"FIELD LENGTH","MAX")</p> <p>I MAX("FIELD LENGTH")>0 S DESC=\$E(DESC,1,MAX("FIELD LENGTH")) ; restriction of the field in the DD</p> <p>;</p> <p>; store the FILENAME, FIELDNAME and VALUE if the APP is IIV and FLN is 350.9. - IB*2.0*506</p> <p>I APP="IIV",FLN=350.9 D Q</p> <p>. S IBCNE=\$G(^IBE(FLN,1,51))</p> <p>. S FRSH=\$P(IBCNE,U,1),TIMOUT=\$P(IBCNE,U,5),RETRY=\$P(IBCNE,U,26)</p> <p>. S DLAYGO=FLN,DIC(0)="L",DIC("DR")="51.01///"_FRSH_"51.05///"_TIMOUT_"51.26///"_RETRY</p> <p>;</p> <p>; add new entry to the table</p> <p>;S DLAYGO=FLN,DIC(0)="L",DIC("DR")=".02///"_DESC</p> <p>S DLAYGO=FLN,DIC(0)="L",DIC("DR")=".02///^S X=DESC"</p> <p>K DD,DO D FILE^DICN K DO</p> <p>Q</p> <p>;</p>	

Routine Name	ERROR^IBCNEHL3	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.2.8	
Related Options	HL7 Processing Incoming RPI Messages	
Related Routines	Routines "Called By"	Routines "Called"
	^IBCNEHL1	
Data Dictionary (DD) References		

Routine Name	ERROR^IBCNEHL3
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
<p>The module of code involved is ERROR^IBCNEHL3:</p> <pre> ; ; ERROR(TQN,ERACT,ERCON,TRCN) ; Entry point ; Input: TQN - IEN for eIV Transmission Queue (#365.1), required ; ERACT - Error Action Code (#365.14), required ; ERCON - Error Condition Code (#365.17), required ; TRCN - Trace # from eIV Response (#365) ; ; ; IIVSTAT - IIV status transmitted by EC ; Note: MAP(IIVSTAT) = IIV STATUS IEN N MSG,ERDESC,ERLEN,XMY,DA,DIE,DR ; ; I \$G(TQN)=" " G ERRORX ; ; ; Scenarios: ; #1 - If error message = "Resubmission Allowed" OR "Please Resubmit ; Original Transaction" - set TQ ; Fut Trans Dt to T + Comm Failure Days and Status to "Hold" I ERACT="R"!(ERACT="P") D G ERRORX . I \$P(\$G(^IBC(365.1,TQN,0)),U,9)=" " D Q ; first time payer asked us to resubmit . . ; Update IIV TQ fields: "Hold" (4), IIV Site Param Comm Failure Days . . D UPDATE(TQN,4,\$P(\$G(^IBE(350.9,1,51)),U,5),ERACT) . . ; . . ; payer asked us to resubmit for the 2nd time for this inquiry . . ; Update IIV TQ fields: "Response Received" (3), n/a ("") . D UPDATE(TQN,3,"",ERACT,ERCON) . ; clear future transmission date so it won't display in the buffer . S DA=TQN,DIE="^IBC(365.1,"DR=".09///@" D ^DIE ; ; ; #2 - If error message = "Please Wait 30 Days and Resubmit" - set TQ ; Fut Trans Dt to T + 30 and Status to "Hold" I ERACT="W" D G ERRORX . ; Update IIV TQ fields: "Hold" (4), 30 . D UPDATE(TQN,4,30,ERACT) ; ; ; #3 - If error message = "Please Wait 10 Days and Resubmit" - set TQ ; Fut Trans Dt to T + 10 and Status to "Hold" I ERACT="X" D G ERRORX . ; Update IIV TQ fields: "Hold" (4), 10 </pre>	

Routine Name	ERROR^IBCNEHL3
<pre> . D UPDATE(TQN,4,10,ERACT) ; ; ; #4 - If error message = "Resubmission Not Allowed" or ; "Do not resubmit" OR "Please correct and resubmit" ; - set TQ Status to "Response Received" ; If we receive error txt, treat as an "N" I ERACT="" S ERACT="N" I ERACT="N"!(ERACT="Y")!(ERACT="S")!(ERACT="C") D G ERRORX ; ; Update IIV TQ fields: "Response Received" (3), n/a ("") . D UPDATE(TQN,3,"",ERACT,ERCON) ; ; ; #5 - Error message is unfamiliar - new Error Action Code ; *** Currently processed in IBCNEHL1 *** ; ; ERRORX ; ERROR exit pt Q ; ; </pre>	
Modified Logic (Changes are in bold)	
<p>Modifications need to be made so that the eIV system doesn't resned an inquiry for any X12 271 message that contains an error action code:</p> <pre> ; ERROR(TQN,ERACT,ERCON,TRCN) ; Entry point ; Input: TQN - IEN for eIV Transmission Queue (#365.1), required ; ERACT - Error Action Code (#365.14), required ; ERCON - Error Condition Code (#365.17), required ; TRCN - Trace # from eIV Response (#365) ; ; ; IIVSTAT - IIV status transmitted by EC ; Note: MAP(IIVSTAT) = IIV STATUS IEN N MSG,ERDESC,ERIEN,XMY,DA,DIE,DR ; ; I \$G(TQN)="" G ERRORX ; ; ; Scenarios: ; #1 - If error message = "Resubmission Allowed" OR "Please Resubmit ; Original Transaction" - set TQ ; Fut Trans Dt to T + Comm Failure Days and Status to "Hold" ; I ERACT="R"!(ERACT="P") D G ERRORX ; . I \$P(\$G(^IBC(365.1,TQN,0)),U,9)="" D Q ; first time payer asked us to resubmit ; . ; Update IIV TQ fields: "Hold" (4), IIV Site Param Comm Failure Days ; . D UPDATE(TQN,4,\$P(\$G(^IBE(350.9,1,51)),U,5),ERACT) ; . ; ; . ; ; . ; payer asked us to resubmit for the 2nd time for this inquiry ; . ; Update IIV TQ fields: "Response Received" (3), n/a ("") ; . D UPDATE(TQN,3,"",ERACT,ERCON) ; . ; clear future transmission date so it won't display in the buffer ; . S DA=TQN,DIE="^IBC(365.1," ,DR=".09///@" D ^DIE ; . ; ; #2 - If error message = "Please Wait 30 Days and Resubmit" - set TQ ; Fut Trans Dt to T + 30 and Status to "Hold" ; I ERACT="W" D G ERRORX </pre>	

Routine Name	ERROR^IBCNEHL3
<pre> ; ; Update IIV TQ fields: "Hold" (4), 30 ; D UPDATE(TQN,4,30,ERACT) ; ; ; #3 - If error message = "Please Wait 10 Days and Resubmit" - set TQ ; Fut Trans Dt to T + 10 and Status to "Hold" ; I ERACT="X" D G ERRORX ; ; Update IIV TQ fields: "Hold" (4), 10 ; D UPDATE(TQN,4,10,ERACT) ; ; ; #4 - If error message = "Resubmission Not Allowed" or ; "Do not resubmit" OR "Please correct and resubmit" ; - set TQ Status to "Response Received" ; If we receive error txt, treat as an "N" ; I \$G(ERACT)=" " S ERACT="N" ; I ERACT="N"!(ERACT="Y")!(ERACT="S")!(ERACT="C") D G ERRORX ; ; Update IIV TQ fields: "Response Received" (3), n/a ("") ; D UPDATE(TQN,3,"",ERACT,ERCON) ; ; ; all scenarios #1 thru #3 & #5 is to be processed exactly like scenario #4 - IB*2*506 D UPDATE(TQN,3,"",ERACT,ERCON) ; ; #5 - Error message is unfamiliar - new Error Action Code ; *** Currently processed in IBCNEHL1 *** ; ; ERRORX ; ERROR exit pt Q ; </pre>	

Routine Name	FIL^IBCNEHL1	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.2.8	
Related Options	HL7 Processing Incoming MFN Messages	
Related Routines	Routines "Called By"	Routines "Called"
		UEACT^IBCNEHL3 RSTA^IBCNEUT7 SST^IBCNEUT2 SAVETQ^IBCNEUT2 SAVFRSH^IBCNEUT5 ERRACT^IBCNEHLU ERROR^IBCNEHL3

Routine Name	FIL^IBCNEHL1	
		RP^IBCNEBF
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local	
Input Attribute Name and Definition		
Output Attribute Name and Definition		
Current Logic		
<p>The module of code involved is FIL^IBCNEHL1:</p> <pre> ; ; FIL ; Finish processing the response message - file into insurance buffer ; ; Input Variables ; ERACT, ERFLG, ERROR, IIVSTAT, MAP, RIEN, TRACE ; ; If no record IEN, quit I \$G(RIEN)="" Q ; N BUFF,DFN,FILEIT,IBFDA,IBIEN,IBQFL,RDAT0,RSRVDT,RSTYPE,SYMBOL,TQDATA,TQN,TQSRVD T ; Initialize variables from the Response File S RDAT0=\$G(^IBCN(365,RIEN,0)),TQN=\$P(RDAT0,U,5) S TQDATA=\$G(^IBCN(365.1,TQN,0)) S IBQFL=\$P(TQDATA,U,11) S DFN=\$P(RDAT0,U,2),BUFF=\$P(RDAT0,U,4) S IBIEN=\$P(TQDATA,U,5),RSTYPE=\$P(RDAT0,U,10) S RSRVDT=\$P(\$G(^IBCN(365,RIEN,1)),U,10) ; ; If an unknown error action or an error filing the response message, ; send a warning email message ; Note - A call to UEACTION will always set ERFLAG=1 I ",W,X,R,P,C,N,Y,S,"[(", "_\$G(ERACT)_,")&(\$G(ERACT)'=")!\$D(ERROR) D UEACTION^IBCNEHL3 ; ; If an error occurred, processing complete I \$G(ERFLG)=1 Q ; ; For an original response, set the Transmission Queue Status to 'Response Received' & ; update remaining retries to comm failure (5) I \$G(RSTYPE)="O" D SST^IBCNEUT2(TQN,3),RSTA^IBCNEUT7(TQN) ; ; Update the TQ service date to the date in the response file </pre>		

Routine Name	FIL^IBCNEHL1
	<pre> ; if they are different AND the Error Action <> ; 'P' for 'Please submit original transaction' ; ; ; *** Temporary change to suppress update of service & freshness dates. ; *** To reinstate, remove comment (;) from next line. ; I TQN="" , \$G(RSTYPE)="O" D ; S TQSRVDT=\$P(\$G(^IBCNEHL(365.1,TQN,0)),U,12) ; I RSRVDT="" , TQSRVDT=RSRVDT, \$G(ERACT)="P" D SAVETQ^IBCNEUT2(TQN,RSRVDT) ; ; update freshness date by same delta ; D SAVFRSH^IBCNEUT5(TQN,+\$FMDIFF^XLFD(TQSRVDT,RSRVDT,1)) ; ; Check for error action I \$G(ERACT)'=""!(\$G(ERTXT)'="") S ERACT=\$\$ERRACT^IBCNEHLU(RIEN),ERCON=\$P(ERACT,U,2),ERACT=\$P(ERACT,U) D ERROR^IBCNEHL3(TQN,ERACT,ERCON,TRACE) G FILX ; ; Stop processing if identification response and not an active policy S FILEIT=1 I \$G(IIVSTAT)=6,TQNJ"" D . I TQDATA="" Q . I IBQFL="I" Q . S FILEIT=0 I 'FILEIT G FILX ; ; If there is an associated buffer entry & one or both of the following ; is true, stop filing (don't update buffer entry) ; 1) buffer status is not 'Entered' ; 2) the buffer entry is verified (* symbol) I BUFF="" , (\$P(\$G(^IBAB(355.33,BUFF,0)),U,4)'="E")!(\$\$SYMBOL^IBCNBLL(BUFF)="") G FILX ; ; Set buffer symbol based on value returned from EC S SYMBOL=MAP(IIVSTAT) ; ; If there is an associated buffer entry, update the buffer entry w/ ; response data I BUFF="" D RP^IBCNEBF(RIEN,"",BUFF) ; ; If no associated buffer entry, create one & populate w/ response ; data (routine call sets IBFDA) I BUFF="" D RP^IBCNEBF(RIEN,1) S BUFF=+IBFDA,UP(365,RIEN_"",.04)=BUFF ; ; Set eIV Processed Date to now S UP(355.33,BUFF_"",.15)=\$\$NOW^XLFD() D FILE^DIE("I","UP","ERROR") FILX ; Q ; </pre>
Modified Logic (Changes are in bold)	
<p>Modifications need to be made so that the eIV system doesn't resned an inquiry for any X12 271 message that contains an error action code:</p> <pre> ; </pre>	

Routine Name	FIL^IBCNEHL1
<pre> FIL ; Finish processing the response message - file into insurance buffer ; ; ; Input Variables ; ERACT, ERFLG, ERROR, IIVSTAT, MAP, RIEN, TRACE ; ; If no record IEN, quit I \$G(RIEN)="" Q ; N BUFF,DFN,FILEIT,IBFDA,IBIEN,IBQFL,RDAT0,RSRVDT,RSTYPE,SYMBOL,TQDATA,TQN,TQSRVD T ; Initialize variables from the Response File S RDAT0=\$G(^IBCNEHL(365,RIEN,0)),TQN=\$P(RDAT0,U,5) S TQDATA=\$G(^IBCNEHL(365.1,TQN,0)) S IBQFL=\$P(TQDATA,U,11) S DFN=\$P(RDAT0,U,2),BUFF=\$P(RDAT0,U,4) S IBIEN=\$P(TQDATA,U,5),RSTYPE=\$P(RDAT0,U,10) S RSRVDT=\$P(\$G(^IBCNEHL(365,RIEN,1)),U,10) ; ; If an unknown error action or an error filing the response message, ; send a warning email message ; Note - A call to UERACT will always set ERFLAG=1 ;I "W,X,R,P,C,N,Y,S,"["_"\$G(ERACT)_"]&(\$G(ERACT)'='')!\$D(ERROR) D UERACT^IBCNEHL3 ; ; If an error occurred, processing complete I \$G(ERFLG)=1 Q ; ; For an original response, set the Transmission Queue Status to 'Response Received' & ; update remaining retries to comm failure (5) I \$G(RSTYPE)="O" D SST^IBCNEUT2(TQN,3),RSTA^IBCNEUT7(TQN) ; ; Update the TQ service date to the date in the response file ; if they are different AND the Error Action <> ; 'P' for 'Please submit original transaction' ; ; *** Temporary change to suppress update of service & freshness dates. ; *** To reinstate, remove comment (;) from next line. ;I TQN="", \$G(RSTYPE)="O" D ; S TQSRVDT=\$P(\$G(^IBCNEHL(365.1,TQN,0)),U,12) ; I RSRVDT="",TQSRVDT=RSRVDT,\$G(ERACT)="P" D SAVETQ^IBCNEUT2(TQN,RSRVDT) ; ; update freshness date by same delta ; D SAVFRSH^IBCNEUT5(TQN,+\$FMDIFF^XLFD(TQSRVDT,RSRVDT,1)) ; ; Check for error action I \$G(ERACT)'=""!(\$G(ERTXT)'='') S ERACT=\$\$ERRACT^IBCNEHLU(RIEN),ERCON=\$P(ERACT,U,2),ERACT=\$P(ERACT,U) D ERROR^IBCNEHL3(TQN,ERACT,ERCON,TRACE) G FILX ; ; Stop processing if identification response and not an active policy S FILEIT=1 I \$G(IIVSTAT)=6,TQN]"" D . I TQDATA="" Q . I IBQFL="I" Q . S FILEIT=0 </pre>	

Routine Name	FIL^IBCNEHL1
<pre> I 'FILEIT G FILX ; ; If there is an associated buffer entry & one or both of the following ; is true, stop filing (don't update buffer entry) ; 1) buffer status is not 'Entered' ; 2) the buffer entry is verified (* symbol) I BUFF="" ,(\$P(\$G(^IBA(355.33,BUFF,0)),U,4)'="E")!(\$\$SYMBOL^IBCNEBLL(BUFF)="") G FILX ; ; Set buffer symbol based on value returned from EC S SYMBOL=MAP(IIVSTAT) ; ; If there is an associated buffer entry, update the buffer entry w/ ; response data I BUFF="" D RP^IBCNEBF(RIEN,"",BUFF) ; ; If no associated buffer entry, create one & populate w/ response ; data (routine call sets IBFDA) I BUFF="" D RP^IBCNEBF(RIEN,1) S BUFF+=IBFDA,UP(365,RIEN_",",.04)=BUFF ; ; Set eIV Processed Date to now S UP(355.33,BUFF_",",.15)=\$\$NOW^XLFD() D FILE^DIE("I","UP","ERROR") FILX ; Q ; </pre>	

Routine Name	IBCNEDEP	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.2.9 thru 2.6.2.13	
Related Options	Insurance Buffer Inquiries	
Related Routines	Routines "Called By"	Routines "Called"
	^IBCNEDE ^IBCNERTQ	MGRP^IBCNEUT5 SST^IBCNEUT2 TMRR^IBCNEDEQ RSTA^IBCNEUT7 CERE^IBCNEDEQ INIT^IBCNEHLO GENERATE^HLMA HLER^IBCNEDEQ SCC^IBCNEDEQ

Routine Name	IBCNEDEP			
		PID^IBCNEHLQ GT1^IBCNEHLQ IN1^IBCNEHLQ NTE^IBCNEHLQ		
Data Dictionary (DD) References				
Related Protocols				
Related Integration Control Registrations (ICRs)				
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local			
Input Attribute Name and Definition				
Output Attribute Name and Definition				
Current Logic				
<pre> IBCNEDEP ;DAOU/ALA - Process Transaction Records ;17-JUN-2002 ;;2.0;INTEGRATED BILLING;**184,271,300,416,438**;21-MAR-94;Build 52 ;;Per VHA Directive 2004-038, this routine should not be modified. ; ; ; This program finds records needing HL7 msg creation ; Periodically check for stop request for background task ; ; ; Variables ; RETR = # retries allowed ; MGRP = Msg Mailgroup ; FAIL = # of days before failure ; FMSG = Failure Mailman flag ; TMSG = Timeout Mailman flag ; FLDT = Failure date ; FUTDT = Future transmission date ; DFN = Patient IEN ; PAYR = Payer IEN ; DTCRT = Date Created ; BUFF = Buffer File IEN ; NRETR = # of retries accomplished ; IHCNT = Count of successful HL7 msgs ; QUERY = Type of msg ; EXT = Which extract produced record ; SRVDT = Service Date ; IRIEN = Insurance Record IEN ; NTRAN = # of transmissions accomplished ; OVRIDE = Override Flag ; BNDL = Bundle Verification Flag ; ; EN ; Entry point </pre>				

Routine Name	IBCNEDEP
	<pre> ; ; ; Start processing of data K ^TMP("HLS",\$J),^TMP("IBQUERY",\$J) ; Initialize count for periodic TaskMan check S IBCNETOT=0 ; ; ; Get IB Site Parameters S IBCNEP=\$G(^IBE(350.9,1,51)) S RETR=+\$P(IBCNEP,U,6),BNDL=\$P(IBCNEP,U,23) S MGRP=\$\$MGRP^IBCNEUT5() S FAIL=\$P(IBCNEP,U,5),TMSG=\$P(IBCNEP,U,7),FMSG=\$P(IBCNEP,U,20) S FLDT=\$\$FMADD^XLFD(DT,-FAIL) ; Statuses ; 1 = Ready To Transmit ; 2 = Transmitted ; 4 = Hold ; 6 = Retry ; HLD ; Go through the 'Hold' statuses, see if ready to be 'retried' S IEN="" F S IEN=\$O(^IBCN(365.1,"AC",4,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) . ; Update count for periodic check . S IBCNETOT=IBCNETOT+1 . ; Check for request to stop background job, periodically . I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q . ; . ; . S FUTDT=\$P(\$G(^IBCN(365.1,IEN,0)),U,9) . ; . ; If the future date is today, set status to 'Retry', . ; DON'T clear future transmission date. (Need date to see if this is the first . ; time that the payer asked us to resubmit this inquiry.) . I FUTDT>DT D SST^IBCNEUT2(IEN,6) ;D . ;. NEW DA,DIE,DR . ;. S DA=IEN,DIE="^IBCN(365.1,"DR=".09///@" D ^DIE . ; ; Exit based on stop request I \$G(ZTSTOP) G EXIT ; TMT ; If the status is 'Transmitted' - is this a 'Retry' or ; 'Comm Failure' S IEN="" F S IEN=\$O(^IBCN(365.1,"AC",2,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) . ; Update count for periodic check . S IBCNETOT=IBCNETOT+1 . ; Check for request to stop background job, periodically . I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q . ; . ; . NEW TDATA,DTCRT,BUFF,DFN,PAYR,XMSUB,VERID . S TDATA=\$G(^IBCN(365.1,IEN,0)) . S DFN=\$P(TDATA,U,2),PAYR=\$P(TDATA,U,3) . S DTCRT=\$P(TDATA,U,6)\1,BUFF=\$P(TDATA,U,5) . S VERID=\$P(TDATA,U,11) . ; ; </pre>

Routine Name	IBCNEDEP
<pre> ; Check against the Failure Date . I DTCRT>FLDT Q ; ; ; If retries are defined . I RETR>0 D Q ; ; Send timeout mail msg .. I PAYR=\$\$FIND1^DIC(365.12,"","X","~NO PAYER") D TMRR^IBCNEDEQ .. D SST^IBCNEUT2(IEN,6) ; ; If no retries defined, set to fail . D SST^IBCNEUT2(IEN,5) ; ; For msg in the Response file set the status to ; 'Comm Failure' . D RSTA^IBCNEUT7(IEN) ; ; Set Buffer symbol to 'B12' (Comm Failure) . I BUFF="" D BUFF^IBCNEUT2(BUFF,15) ; . I PAYR=\$\$FIND1^DIC(365.12,"","X","~NO PAYER") Q ; ; Issue comm fail MailMan msg only for ver'ns . I VERID="V" D CERR^IBCNEDEQ ; ; Exit for stop request I \$G(ZTSTOP) G EXIT ; RET ; If status is 'Retry' S IEN="" F S IEN=\$O(^IBCN(365.1,"AC",6,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) ; Update count for periodic check . S IBCNETOT=IBCNETOT+1 ; Check for request to stop background job, periodically . I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q ; . NEW TDATA,NRETR,PAYR,BUFF,DFN,MSG,RIEN,HIEN,XMSUB,VERID . S TDATA=\$G(^IBCN(365.1,IEN,0)) . S NRETR=\$P(TDATA,U,8),PAYR=\$P(TDATA,U,3) . S BUFF=\$P(TDATA,U,5),DFN=\$P(TDATA,U,2) . S VERID=\$P(TDATA,U,11) . S NRETR=NRETR+1 ; ; If retries are finished, set to fail . I NRETR>RETR D Q .. D SST^IBCNEUT2(IEN,5) ; ; Set Buffer symbol to 'B12' (Comm Failure) .. I BUFF="" D BUFF^IBCNEUT2(BUFF,15) ; ; For msg in the Response file set the status to ; 'Comm Failure' .. D RSTA^IBCNEUT7(IEN) </pre>	

Routine Name	IBCNEDEP
<pre> .. I PAYR=\$\$FIND1^DIC(365.12,"","X","~NO PAYER") Q .. ; .. I VERID="V" D CERE^IBCNEDEQ ; ; If generating retry, set elV status to comm failure (5) for ; ; remaining related responses . D RSTA^IBCNEUT7(IEN) ; ; Exit for stop request I \$G(ZTSTOP) G EXIT ; FIN ; Prioritize requests for statuses 'Retry' and 'Ready to Transmit' ; ; Separate inquiries into verifications, identifications, ; and "fishes" - VNUM = Priority of output F STA=1,6 S IEN="" D . F S IEN=\$O(^IBCN(365.1,"AC",STA,IEN)) Q:IEN="" D .. S IBDATA=\$G(^IBCN(365.1,IEN,0)) Q:IBDATA="" .. S QUERY=\$P(IBDATA,U,11),DFN=\$P(IBDATA,U,2),OVRIDE=\$P(IBDATA,U,14) .. S PAYR=\$P(IBDATA,U,3) .. I QUERY="V" S VNUM=3 .. I QUERY="V" D ... I PAYR=\$\$FIND1^DIC(365.12,"X","~NO PAYER") S VNUM=5 Q ... S VNUM=4 .. I OVRIDE="" D ... I PAYR=\$\$FIND1^DIC(365.12,"X","~NO PAYER") S VNUM=2 Q ... S VNUM=1 .. S ^TMP("IBQUERY",\$J,VNUM,DFN,IEN)="" ; LP ; Loop through priorities, process as either verifications ; or identifications N IHCNT S VNUM="",IHCNT=0 F S VNUM=\$O(^TMP("IBQUERY",\$J,VNUM)) Q:VNUM="" D Q:\$G(ZTSTOP)!\$G(QFL)=1 . I VNUM=1!(VNUM=3) D VER Q . ;D ID ; EXIT ; Finish K BUFF,CNT,D,D0,DA,DFN,DI,DIC,DIE,DISYS,DQ,DR,DTCRT,EXT,FAIL,FLDT,FUTDT K FRDT,FMSG,GT1,HCT,HIEN,HL,HLCDOM,HLCINS,HLCS,HLCSTCP,HLDOM,HLECH,%I,%H K HLEID,HLFS,HLHDR,HLINST,HLIP,HLN,HLPARAM,HLPROD,HLQ,HLRESLT,XMSUB K HLSAN,HLTYPE,HLX,IBCNEP,IBCNHLP,IEN,IHCNT,IN1,IRIEN,MDTM,MGRP,MSGID,TOT K NRETR,NTRAN,OVRIDE,PAYR,PID,QFL,QUERY,RETR,RSIEN,SRVDT,STA,TRANSR,X K ZMID,^TMP("IBQUERY",\$J),Y,DOD,DGREL,TMSG,RSTYPE,OMSGID,QFL K IBCNETOT,HLP,SUBID,VNUM,BNDL,IBDATA,PATID Q ; VER ; Initialize HL7 variables protocol for Verifications S IBCNHLP="IBCNE IIV RQV OUT" D INIT^IBCNEHLO ; S DFN="" F S DFN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN)) Q:DFN="" D Q:\$G(ZTSTOP) . ; </pre>	

Routine Name	IBCNEDEP
	<pre> . ; If the INQUIRE SECONDARY INSURANCES flag is 'yes', . ; bundle verifications together, send a continuation pointer . I VNUM=3,BNDL D Q:QFL .. S TOT=0,IEN="",QFL=0 .. F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" S TOT=TOT+1 . ; . S IEN="",OMSGID="",QFL=0,CNT=0 . F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) .. ; Update count for periodic check .. S IBCNETOT=IBCNETOT+1 .. ; Check for request to stop background job, periodically .. I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q .. ; .. D PROC I PID="" Q .. ; .. I BNDL S HLP("CONTPTR")=\$G(OMSGID) .. ; D GENERATE^HLMA(HLEID,"GM",1,.HLRESLT,"",.HLP) .. D GENERATE^HLMA(IBCNEHLP,"GM",1,.HLRESLT,"",.HLP) .. K ^TMP("HLS",\$J),HLP .. ; .. ; If not successful .. I \$P(HLRESLT,U,2)]"" D HLER^IBCNEDEQ Q .. ; If successful .. D SCC^IBCNEDEQ .. I BNDL D ... I CNT=1 S OMSGID=MSGID . ; . K HL,IN1,GT1,PID,DFN,^TMP(\$J,"HLS") Q . ; ID ; Send Identification Msgs . ; . ; Initialize the HL7 variables based on the HL7 protocol S IBCNEHLP="IBCNE IIV RQI OUT" D INIT^IBCNEHLO . ; S DFN="" F S DFN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN)) Q:DFN="" D Q:\$G(ZTSTOP)!QFL . ; Update count for periodic check . S IBCNETOT=IBCNETOT+1 . ; Check for request to stop background job, periodically . I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q . ; . S TOT=0,IEN="",CNT=0,OMSGID="",QFL=0 . ; . ; Get the total # of identification msgs for a patient . F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" S TOT=TOT+1 . ; . ; For each identification transaction generate an HL7 msg . F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" D .. D PROC .. ; .. I VNUM=4 S HLP("CONTPTR")=\$G(OMSGID) </pre>

Routine Name	IBCNEDEP
	<pre> .. ; D GENERATE^HLMA(HLEID,"GM",1,.HLRESLT,"",.HLP) .. D GENERATE^HLMA(IBCNEHLQ,"GM",1,.HLRESLT,"",.HLP) .. K ^TMP("HLS",\$J),HLP .. ; .. ; If not successful .. I \$P(HLRESLT,U,2)]"" D HLER^IBCNEDEQ Q .. ; .. ; If successful .. D SCC^IBCNEDEQ .. I VNUM=4 D ... I CNT=1 S OMSGID=MSGID ; Q ; PROC ; Process TQ record S TRANSR=\$G(^IBCN(365.1,IEN,0)) S DFN=\$P(TRANSR,U,2),PAYR=\$P(TRANSR,U,3),BUFF=\$P(TRANSR,U,5) S QUERY=\$P(TRANSR,U,11),EXT=\$P(TRANSR,U,10),SRVDT=\$P(TRANSR,U,12) S IRIEN=\$P(TRANSR,U,13),HCT=0,NTRAN=\$P(TRANSR,U,7),NRETR=\$P(TRANSR,U,8) S SUBID=\$P(TRANSR,U,16),OVERRIDE=\$P(TRANSR,U,14),STA=\$P(TRANSR,U,4) S FRDT=\$P(TRANSR,U,17),PATID=\$P(TRANSR,U,19) ; ; Build the HL7 msg S HCT=HCT+1,^TMP("HLS",\$J,HCT)="PRD NA" D PID^IBCNEHLQ I PID=""!(PID?."") Q S HCT=HCT+1,^TMP("HLS",\$J,HCT)=\$TR(PID,"*",") D GT1^IBCNEHLQ I GT1=""!GT1?."*" S HCT=HCT+1,^TMP("HLS",\$J,HCT)=\$TR(GT1,"*",") D IN1^IBCNEHLQ I IN1=""!IN1?."*" D . S HCT=HCT+1 . I VNUM=1 S ^TMP("HLS",\$J,HCT)=\$TR(IN1,"*",") Q . I VNUM=2,BNDL S ^TMP("HLS",\$J,HCT)=\$TR(IN1,"*",") Q . S CNT=CNT+1 I TOT=0 S TOT=1 . S \$P(IN1,HLFS,22)=TOT,\$P(IN1,HLFS,21)=CNT . S ^TMP("HLS",\$J,HCT)=\$TR(IN1,"*",") ; ; Build multi-field NTE segment D NTE^IBCNEHLQ ; If build successful I NTE=""!\$E(NTE,1)="" S HCT=HCT+1,^TMP("HLS",\$J,HCT)=\$TR(NTE,"*",") K NTE Q </pre>
Modified Logic (Changes are in bold)	
	<p>Modifications to be made to the ^IBCNEDEP routine for requirements 2.6.2.9 thru 2.6.2.13:</p> <p>IBCNEDEP ;DAOU/ALA - Process Transaction Records ;17-JUN-2002 ;;2.0;INTEGRATED BILLING;184,271,300,416,438;21-MAR-94;Build 52 ;;Per VHA Directive 2004-038, this routine should not be modified. ; ; This program finds records needing HL7 msg creation</p>

Routine Name	IBCNEDEP
<pre> ; Periodically check for stop request for background task ; ; ; Variables ; RETR = # retries allowed ; MGRP = Msg Mailgroup ; FAIL = # of days before failure ; FMSG = Failure Mailman flag ; TMSG = Timeout Mailman flag ; FLDT = Failure date ; FUTDT = Future transmission date ; DFN = Patient IEN ; PAYR = Payer IEN ; DTCRT = Date Created ; BUFF = Buffer File IEN ; NRETR = # of retries accomplished ; IHCNT = Count of successful HL7 msgs ; QUERY = Type of msg ; EXT = Which extract produced record ; SRVDT = Service Date ; IRIEN = Insurance Record IEN ; NTRAN = # of transmissions accomplished ; OVERRIDE = Override Flag ; BNDL = Bundle Verification Flag ; EN ; Entry point ; ; Start processing of data K ^TMP("HLS",\$J),^TMP("IBQUERY",\$J) ; Initialize count for periodic TaskMan check S IBCNETOT=0 ; ; Get IB Site Parameters S IBCNEP=\$G(^IBE(350.9,1,51)) S RETR=+\$P(IBCNEP,U,6),BNDL=\$P(IBCNEP,U,23) S MGRP=\$\$MGRP^IBCNEUT5() S FAIL=\$P(IBCNEP,U,5),TMSG=\$P(IBCNEP,U,7),FMSG=\$P(IBCNEP,U,20) S RETRYFLG=\$P(IBCNEP,U,26) ; IB*2.0*506 S FLDT=\$\$FMADD^XLFD(TD,-FAIL) ; Statuses ; 1 = Ready To Transmit ; 2 = Transmitted ; 4 = Hold ; 6 = Retry ; ; If the status is 'HOLD' is this a 'Retry'? ; DO HLD ; this is not to be called unless the status of HOLD is reinstated .. see HLD tag ; below ; and the code within ERROR^IBCNEHL3 ; ; Exit based on stop request I \$G(ZTSTOP) G EXIT ; TMT ; If the status is 'Transmitted' - is this a 'Retry' or </pre>	

Routine Name	IBCNEDEP
<pre> ; 'Comm Failure' S IEN="" F S IEN=\$O(^IBCN(365.1,"AC",2,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) ; ; Update count for periodic check . S IBCNETOT=IBCNETOT+1 ; ; Check for request to stop background job, periodically . I \$D(ZTQUEUED),IBCN(365.1,IEN)=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q ; ; . NEW TDATA,DTCRT,BUFF,DFN,PAYR,XMSUB,VERID . S TDATA=\$G(^IBCN(365.1,IEN,0)) . S DFN=\$P(TDATA,U,2),PAYR=\$P(TDATA,U,3) . S DTCRT=\$P(TDATA,U,6)\1,BUFF=\$P(TDATA,U,5) . S VERID=\$P(TDATA,U,11) ; ; ; ; Check against the Failure Date . I DTCRT>FLDT Q ; ; ; ; If retries are allowed ; ; I RETR>0 D Q ; comment out ; ; I RETRYFLG="Y" D Q ; ; ; ; Send timeout mail msg ; ; I PAYR=\$\$FIND1^DIC(365.12,"X","~NO PAYER") D TMRR^IBCNEDEQ ; comment out ; ; ; ; I \$\$PYRACTV^IBCNEDE7(PAYR) Q ; If Payer is not Nationally Active skip record ; ; ; ; D SST^IBCNEUT2(IEN,6) ; mark TQ entry status as 'retry' ; ; ; ; ; ; D SST^IBCNEUT2(IEN,5) ; if RETRYFLAG=NO set TQ record to 'communication failure' ; ; ; ; For msg in the Response file set the status to ; ; 'Comm Failure' . D RSTA^IBCNEUT7(IEN) ; ; ; ; Set Buffer symbol to 'C1' (Comm Failure) ; used to be "B12" - ien of 15 ; ; I BUFF="" D BUFF^IBCNEUT2(BUFF,15) ; symbol no longer used - IB*2.0*506 ; ; I BUFF="" D BUFF^IBCNEUT2(BUFF,12) ; set to "#" communication failure ; ; ; ; I PAYR=\$\$FIND1^DIC(365.12,"X","~NO PAYER") Q ; ; ; ; Issue comm fail MailMan msg only for ver'ns ; ; I VERID="V" D CERR^IBCNEDEQ ; removed IB*2.0*506 ; ; ; ; Exit for stop request I \$G(ZTSTOP) G EXIT ; ; RET ; If status is 'Retry' ; retries only exist if the RETRYFLG = YES S IEN="" F S IEN=\$O(^IBCN(365.1,"AC",6,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) ; ; Update count for periodic check . S IBCNETOT=IBCNETOT+1 ; ; Check for request to stop background job, periodically </pre>	

Routine Name	IBCNEDEP
<pre> . I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$\$^%ZTLOAD() S ZTSTOP=1 Q .; . NEW TDATA,NRETR,PAYR,BUFF,DFN,MSG,RIEN,HIEN,XMSUB,VERID . S TDATA=\$G(^IBCN(365.1,IEN,0)) . S NRETR=\$P(TDATA,U,8),PAYR=\$P(TDATA,U,3) . S BUFF=\$P(TDATA,U,5),DFN=\$P(TDATA,U,2) . S VERID=\$P(TDATA,U,11) . S NRETR=NRETR+1 .; .; If retries are finished, set to communication failure . I NRETR>RETR D Q .. D SST^IBCNEUT2(IEN,5) .. ; .. ; Set Buffer symbol to 'C1' (Comm Failure) ; used to be "B12" – ien of 15 .. I BUFF="" D BUFF^IBCNEUT2(BUFF,15) ; symbol no longer used – IB*2.0*506 .. I BUFF="" D BUFF^IBCNEUT2(BUFF,12) ; set to "#" communication failure .. ; .. ; For msg in the Response file set the status to .. ; 'Comm Failure' .. D RSTA^IBCNEUT7(IEN) .. I PAYR=\$\$FIND1^DIC(365.12,"","X","~NO PAYER") Q .. ; .. I VERID="V" D CERE^IBCNEDEQ ;removed IB*2.0*506 .; If generating retry, set eIV status to comm failure (5) for .; remaining related responses . D RSTA^IBCNEUT7(IEN) .; .; Exit for stop request I \$G(ZTSTOP) G EXIT .; FIN ; Prioritize requests for statuses 'Retry' and 'Ready to Transmit' .; .; Separate inquiries into verifications, identifications, .; and "fishes" - VNUM = Priority of output F STA=1,6 S IEN="" D . F S IEN=\$O(^IBCN(365.1,"AC",STA,IEN)) Q:IEN="" D .. S IBDATA=\$G(^IBCN(365.1,IEN,0)) Q:IBDATA="" .. S QUERY=\$P(IBDATA,U,11),DFN=\$P(IBDATA,U,2),OVRIDE=\$P(IBDATA,U,14) .. S PAYR=\$P(IBDATA,U,3) .. I QUERY="V" S VNUM=3 .. I QUERY="V" D ... I PAYR=\$\$FIND1^DIC(365.12,"X","~NO PAYER") S VNUM=5 Q ... S VNUM=4 .. I OVRIDE="" D ... I PAYR=\$\$FIND1^DIC(365.12,"X","~NO PAYER") S VNUM=2 Q ... S VNUM=1 .. S ^TMP("IBQUERY",\$J,VNUM,DFN,IEN)="" .; LP ; Loop through priorities, process as either verifications .; or identifications N IHCNT S VNUM="",IHCNT=0 F S VNUM=\$O(^TMP("IBQUERY",\$J,VNUM)) Q:VNUM="" D Q:\$G(ZTSTOP)!\$G(QFL)=1 </pre>	

Routine Name	IBCNEDEP
<pre> . I VNUM=1!(VNUM=3) D VER Q .;D ID ; EXIT ; Finish K BUFF,CNT,D,D0,DA,DFN,DI,DIC,DIE,DISYS,DQ,DR,DTCRT,EXT,FAIL,FLDT,FUTDT K FRDT,FMSG,GT1,HCT,HIEN,HL,HLCDOM,HLCINS,HLCS,HLCSTCP,HLDOM,HLECH,%I,%H K HLEID,HLFS,HLHDR,HLINST,HLIP,HLN,HLPARAM,HLPROD,HLQ,HLRESLT,XMSUB K HLSAN,HLTYPE,HLX,IBCNEP,IBCNHLP,IEN,IHCNT,IN1,IRIEN,MDTM,MGRP,MSGID,TOT K NRETR,NTRAN,OVERRIDE,PAYR,PID,QFL,QUERY,RETR,RSIEN,SRVDT,STA,TRANSR,X K ZMID,^TMP("IBQUERY",\$J),Y,DOD,DGREL,TMSG,RSTYPE,OMSGID,QFL K IBCNETOT,HLP,SUBID,VNUM,BNDL,IBDATA,PATID,RETRYFLG Q ; VER ; Initialize HL7 variables protocol for Verifications S IBCNHLP="IBCNE IIV RQV OUT" D INIT^IBCNEHLO ; S DFN="" F S DFN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN)) Q:DFN="" D Q:\$G(ZTSTOP) .; .; If the INQUIRE SECONDARY INSURANCES flag is 'yes', .; bundle verifications together, send a continuation pointer . I VNUM=3,BNDL D Q:QFL .. S TOT=0,IEN="",QFL=0 .. F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" S TOT=TOT+1 .; . S IEN="",OMSGID="",QFL=0,CNT=0 . F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) .. ; Update count for periodic check .. S IBCNETOT=IBCNETOT+1 .. ; Check for request to stop background job, periodically .. I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q .. ; .. D PROC I PID="" Q .. ; .. I BNDL S HLP("CONTPTR")=\$G(OMSGID) .. ; D GENERATE^HLMA(HLEID,"GM",1,.HLRESLT,"",.HLP) .. D GENERATE^HLMA(IBCNHLP,"GM",1,.HLRESLT,"",.HLP) .. K ^TMP("HLS",\$J),HLP .. ; .. ; If not successful .. I \$P(HLRESLT,U,2)]"" D HLER^IBCNEDEQ Q .. ; If successful .. D SCC^IBCNEDEQ .. I BNDL D ... I CNT=1 S OMSGID=MSGID ; K HL,IN1,GT1,PID,DFN,^TMP(\$J,"HLS") Q ; ID ; Send Identification Msgs ; ; Initialize the HL7 variables based on the HL7 protocol </pre>	

Routine Name	IBCNEDEP
<pre> S IBCNHLP="IBCNE IIV RQI OUT" D INIT^IBCNEHLO ; S DFN="" F S DFN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN)) Q:DFN="" D Q:\$G(ZTSTOP)!QFL ; ; Update count for periodic check . S IBCNETOT=IBCNETOT+1 ; ; Check for request to stop background job, periodically . I \$D(ZTQUEUED),IBCNETOT#100=0,\$\$S^%ZTLOAD() S ZTSTOP=1 Q ; . S TOT=0,IEN="",CNT=0,OMSGID="",QFL=0 ; ; ; Get the total # of identification msgs for a patient . F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" S TOT=TOT+1 ; ; ; For each identification transaction generate an HL7 msg . F S IEN=\$O(^TMP("IBQUERY",\$J,VNUM,DFN,IEN)) Q:IEN="" D .. D PROC .. ; .. I VNUM=4 S HLP("CONTPTR")=\$G(OMSGID) .. ; D GENERATE^HLMA(HLEID,"GM",1,.HLRESLT,"",.HLP) .. D GENERATE^HLMA(IBCNEHLP,"GM",1,.HLRESLT,"",.HLP) .. K ^TMP("HLS",\$J),HLP .. ; .. ; If not successful .. I \$P(HLRESLT,U,2)]"" D HLER^IBCNEDEQ Q .. ; .. ; If successful .. D SCC^IBCNEDEQ .. I VNUM=4 D ... I CNT=1 S OMSGID=MSGID ; Q ; PROC ; Process TQ record S TRANSR=\$G(^IBCN(365.1,IEN,0)) S DFN=\$P(TRANSR,U,2),PAYR=\$P(TRANSR,U,3),BUFF=\$P(TRANSR,U,5) S QUERY=\$P(TRANSR,U,11),EXT=\$P(TRANSR,U,10),SRVDT=\$P(TRANSR,U,12) S IRIEN=\$P(TRANSR,U,13),HCT=0,NTRAN=\$P(TRANSR,U,7),NRETR=\$P(TRANSR,U,8) S SUBID=\$P(TRANSR,U,16),OVERRIDE=\$P(TRANSR,U,14),STA=\$P(TRANSR,U,4) S FRDT=\$P(TRANSR,U,17),PATID=\$P(TRANSR,U,19) ; ; Build the HL7 msg S HCT=HCT+1,^TMP("HLS",\$J,HCT)="PRD NA" D PID^IBCNEHLQ I PID=""!(PID?."") Q S HCT=HCT+1,^TMP("HLS",\$J,HCT)=\$TR(PID,"*",",") D GT1^IBCNEHLQ I GT1=""!GT1?."*" S HCT=HCT+1,^TMP("HLS",\$J,HCT)=\$TR(GT1,"*",",") D IN1^IBCNEHLQ I IN1=""!IN1?."*" D . S HCT=HCT+1 . I VNUM=1 S ^TMP("HLS",\$J,HCT)=\$TR(IN1,"*",",") Q . I VNUM=2,BNDL S ^TMP("HLS",\$J,HCT)=\$TR(IN1,"*",",") Q . S CNT=CNT+1 I TOT=0 S TOT=1 . S \$P(IN1,HLFS,22)=TOT,\$P(IN1,HLFS,21)=CNT </pre>	

Routine Name	IBCNEDEP
<pre> . S ^TMP("HLS",\$J,HCT)=\$TR(IN1,"*", "") ; ; Build multi-field NTE segment D NTE^IBCNEHLQ ; If build successful I NTE="" , \$E(NTE,1)'="" S HCT=HCT+1, ^TMP("HLS",\$J,HCT)=\$TR(NTE,"*", "") K NTE Q ; ; The tag HLD was found at the top of this routine. It was moved ; to its own procedure because we don't need it anymore at this time. ; Responses will not have the status of HOLD starting with patch IB*2*506. ; If HOLD is reinstated, then the logic below must be rewritten for the ; appropriate retry logic at that time. HLD ; Go through the 'Hold' statuses, see if ready to be 'retried' Q ;Quit added as safety valve ;S IEN="" ;F S IEN=\$O(^IBCNC(365.1,"AC",4,IEN)) Q:IEN="" D Q:\$G(ZTSTOP) ; ; Update count for periodic check ; S IBCNETOT=IBCNETOT+1 ; ; Check for request to stop background job, periodically ; I \$D(ZTQUEUED), IBCNETOT#100=0, \$\$S^%ZTLOAD() S ZTSTOP=1 Q ; ; S FUTDT=\$P(\$G(^IBCNC(365.1,IEN,0)),U,9) ; ; ; If the future date is today, set status to 'Retry', ; ; DON'T clear future transmission date. (Need date to see if this is the first ; ; time that the payer asked us to resubmit this inquiry.) ; ; I FUTDT>DT D S S T^IBCNEUT2(IEN,6) ;D ; comment out ; ; NEW DA,DIE,DR ; ; S DA=IEN,DIE="^IBCNC(365.1,"DR=".09///@" D ^DIE ; ; I FUTDT>DT D ; ; D S S T^IBCNEUT2(IEN,6) ; set TQ status to 'retry' Q </pre>	

3.3.3 Data Dictionaries

File Name and Number	IIV Status Table (#365.15)
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Requirements Traceability Matrix	2.6.2.13
Related Options	
Data Dictionary (DD) References	

Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
File Documentation	
File Auditing, Security, and Archiving	

Field Name	CODE
Field Description	Code
Field #	.01
Node #	0
Piece #	1
New Field	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Data Type	<input type="checkbox"/> Date/Time <input type="checkbox"/> Numeric <input type="checkbox"/> Set of Codes <input type="checkbox"/> Free Text <input checked="" type="checkbox"/> Pointer to a File <input type="checkbox"/> Variable-Pointer
Identifier	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Uneditable Field	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Mandatory Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Field Documentation or Help Changes Necessary	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Field Definition	
Input/Output Transform	Edit this field so that the <u>OUTPUT TRANSFORM</u> includes the following in addition to what is already there ... \$E(Y)="C": "Problem Identified, Communication Failure"
Cross-Reference (id and type) No cross reference	<input type="checkbox"/> Regular <input type="checkbox"/> Kwic <input type="checkbox"/> Mnemonic <input type="checkbox"/> Mumps <input type="checkbox"/> Soundex <input type="checkbox"/> Trigger <input type="checkbox"/> Bulletin

File Name and Number	IB SITE PARAMETERS (#350.9)
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Requirements Traceability Matrix	2.6.2.3
Related Options	MCCR Site Parameter Display Edit [IBJ MCCR SITE PARAMETER]

Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
File Documentation	
File Auditing, Security, and Archiving	

Field Name	RETRY FLAG
Field Description	RETRY FLAG
Field #	350.9
Node #	51
Piece #	26
New Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Data Type	<input type="checkbox"/> Date/Time <input type="checkbox"/> Numeric <input checked="" type="checkbox"/> Set of Codes <input type="checkbox"/> Free Text <input type="checkbox"/> Pointer to a File <input type="checkbox"/> Variable-Pointer
Identifier	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Uneditable Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Mandatory Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Field Documentation or Help Changes Necessary	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Field Definition	Flag that indicates whether VistA is authorized to retransmit an eIV Inquiry if no response is received within the number of TimeOut Days (#350.9,51.05) This data value is stored by the TFIL^IBCNEHLT routine.
Input/Output Transform	
Cross-Reference (id and type)	<input type="checkbox"/> Regular <input type="checkbox"/> Kwic <input type="checkbox"/> Mnemonic <input type="checkbox"/> Mumps <input type="checkbox"/> Soundex <input type="checkbox"/> Trigger <input type="checkbox"/> Bulletin
No cross reference	

3.4 System Feature: Enhancements to the eIV Site Parameters

This section covers the design for the requirements listed in Section 2.6.3 of the RSD.

3.4.1 Functional Requirements:

Table 3: Enhancements to the eIV Site Parameters

SRSRSD REQ ID	REQ Title	Comments / Notes
2.6.3.1	eIV Site Parameters – Retry Flag Not Editable	No code changes necessary
2.6.3.2	eIV Site Parameters - Freshness Days Not Editable	
2.6.3.3	eIV Site Parameters – Timeout Days Not Editable	No code changes necessary
2.6.3.4	eIV Site Parameters – Set the Value of “# of Retries” Field	
2.6.3.5	eIV Site Parameters – Set the Initial Value of the Retry Flag	
2.6.3.6	eIV Site Parameters – Set the Initial Value of the Freshness Days	
2.6.3.7	eIV Site Parameters – Set the Initial Value of the Timeout Days	
2.6.3.8	eIV Site Parameters – Set the Value of the 'HL7 Response Processing' Field	
2.6.3.9	eIV Site Parameters – 'HL7 Response Processing' Field Not Editable	
2.6.3.10	eIV Site Parameters – Restrict eIV Number of Possible Retries	

3.4.2 Routines

Routine Name	IBY506PO	
Enhancement Category	<input checked="" type="checkbox"/> New <input type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.3.4 – 2.6.3.8 and 2.6.3.10 Also, 2.6.2.8 thru 2.6.2.13	
Related Options	Post Installation	
Related Routines	Routines “Called By”	Routines “Called”
	Installation of KIDS build	SST^IBCNEUT2 RSTA^IBCNEUT7 BUFF^IBCNEUT2
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control		

Routine Name	IBY506PO
Registrations (ICRs)	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
No Current Logic available...this is a new routine.	
Modified Logic (Changes are in bold)	
<p>1) The Post Installation routine ^IBY506PO will require code to populate the fields mentioned in the requirements for 2.6.3.4 thru 2.6.3.8 and 2.6.3.10:</p> <ul style="list-style-type: none"> - The "# of Retries" eIV Site Parameter (in #350.9, 51.06) will be set to an initial value of "1". - The "Retry Flag" eIV Site Parameter (in #350.9, 51.26) will be to an initial value of "0" (zero) for "NO". - The "Freshness Days" eIV Site Parameter (in #350.9, 51.01) will be to an initial value of "180". - The "Timeout Days" eIV Site Parameter (in #350.9, 51.05) will be to an initial value of "5". - The "HL7 Response Processing" eIV Site Parameter (in #350.9, 51.13) will be to an initial value of "I" for "IMMEDIATE". <p>2) The Post Installation routine ^IBY506PO will need to have the following code to help satisfy the requirement 2.6.2.8:</p> <p>Loop though all TQ entries that have a status of HOLD and mark them as communication failure.</p> <p>NEW IEN,BUFF</p> <pre> S IEN="" F S IEN=\$O(^IBCNEUT2(IEN,5)) D Q:IEN="" . D SST^IBCNEUT2(IEN,5) ; set TQ record to 'communication failure' . ; . ; For msg in the Response file set the status to 'Comm Failure' . D RSTA^IBCNEUT7(IEN) . ; . ; Set Buffer symbol to 'C1' (Comm Failure) . S BUFF=\$P(\$G(^IBCNEUT2(IEN,5)),U,5) . I BUFF="" D BUFF^IBCNEUT2(BUFF,12) ; set to "#" communication failure Q </pre>	

Routine Name	IBY506PO
<p>3) The Post Installation routine ^IBY506PO will require code to create the following two new entry in the IIV STATUS TABLE file #365.15 to satisfy the requirements 2.6.2.13 & 2.6.1.22 (respectively):</p> <p>Code = "C1" ASCII Value for IIV status = 35 Expand Entry Action = 0 Description ="eIV was unable to electronically verify this insurance information due to a communication failure." Corrective Action = "Action to take: Contact the insurance company to manually verify this insurance information."</p> <p>The "C1" entry is almost identical to 'B12'. Same description and corrective action; however, we needed the symbol and output of the code to be different than if it was a "!" type of error. I didn't want to remove or alter the existing B12 record since it was used in past records.</p> <p>Code = "E1" ASCII Value for IIV status = 36 Expand Entry Action = EE Update is Not Allowed Description =" Information received via electronic inquiry indicates patient has active insurance; however another verify did not have the ability to process this entry." Corrective Action = " Action to take: Review the details listed in the eIV Response Report before processing this buffer entry."</p>	

3.4.3 Templates

Template Name	IBCNE GENERAL PARAMETER EDIT	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirements Traceability Matrix	2.6.3.2, 2.6.3.9	
Template Type	<input type="checkbox"/> Sort <input checked="" type="checkbox"/> Input <input type="checkbox"/> Print <input type="checkbox"/> Other	
Related Options		
Related Routines	Routines "Called By"	Routines "Called"
Data Dictionary (DD) References		
Global References		

NUMBER: 1667	NAME: IBCNE GENERAL PARAMETER EDIT
DATE CREATED: JUL 03, 2013@12:32	READ ACCESS: @
FILE: IB SITE PARAMETERS	USER #: 123456991
WRITE ACCESS: @	DATE LAST USED: JUL 03, 2013

Before:

```

EDIT FIELDS (c): FRESHNESS DAYS
EDIT FIELDS (c): MESSAGES MAILGROUP
EDIT FIELDS (c): HL7 RESPONSE PROCESSING
EDIT FIELDS (c): S:X="I" Y="@1"
EDIT FIELDS (c): HL7 START TIME
EDIT FIELDS (c): HL7 STOP TIME
EDIT FIELDS (c): @1
EDIT FIELDS (c): CONTACT PERSON
EDIT FIELDS (c): CONTACT PERSON:
EDIT FIELDS (c): OFFICE PHONE;REQ
EDIT FIELDS (c): EMAIL ADDRESS;REQ
EDIT FIELDS (c): FAILURE MAILMAN MSG    COMPILED (c): NO

```

After:

```

EDIT FIELDS (c): MESSAGES MAILGROUP
EDIT FIELDS (c): S:X="I" Y="@1"
EDIT FIELDS (c): HL7 START TIME
EDIT FIELDS (c): HL7 STOP TIME
EDIT FIELDS (c): @1
EDIT FIELDS (c): CONTACT PERSON
EDIT FIELDS (c): CONTACT PERSON:
EDIT FIELDS (c): OFFICE PHONE;REQ
EDIT FIELDS (c): EMAIL ADDRESS;REQ
EDIT FIELDS (c): FAILURE MAILMAN MSG    COMPILED (c): NO

```

3.5 System Feature: Enhancements using Security Keys

This section covers the design for the requirements listed in Section 2.6.4 of the RSD.

3.5.1 Functional Requirements:

Table 4: Enhancements using Security Keys

SR S RSD REQ ID	REQ Title	Comments / Notes
2.6.4.1	Security Key – Create New Key to Add/Edit an Insurance Company	

<u>SRSRSD</u> REQ ID	REQ Title	Comments / Notes
2.6.4.2	Security Key – Lock the “Insurance Company Entry/Edit” Option	
2.6.4.3	Security Key – User Requires Key to Add/Edit Insurance Company in the Insurance Buffer	
2.6.4.4	Security Key – Create New Key to Add/Edit a Group/Plan	
2.6.4.5	Security Key – User Requires Key to Add/Edit Group/Plan in the Buffer	
2.6.4.6	Security Key – Lock the Ability to Create a Group/Plan within 'Patient Insurance Info View/Edit' option	

This section also addresses Requirements 2.6.1.16 & 2.6.1.17 (Table 1: Enhancements to the Insurance Buffer)

- Requirement 2.6.1.16: Insurance Buffer – Remove Ability to Create New Insurance Company
- Requirement 2.6.1.17: Insurance Buffer – Remove Ability to Create New Group/Plan

Below is a high level overview of the processing logic concerning implementation of two new Security Keys for SRSRSD REQ IDs: 2.6.4.3 and 2.6.4.6 with respect to the behavior of the Insurance Buffer (action process/accept) for SRSRSD REQ IDs 2.6.1.16 & 2.6.1.17. Detailed logic is stated in the Routine section 3.5.2.

SRSRSD Requirement ID: 2.6.1.16

Institute a new Security Key for Insurance Company Edit (IBCN INSURANCE COMPANY EDIT)

1. If Key exists the code will behave as the code currently does.
2. If the Key doesn't exist then:
 - a. If this the correct Insurance Company then go to Group/Plan prompt
 - b. If this isn't the correct Insurance Company then display the message that “A Supervisor will need to add the Insurance Company before processing can continue”. Then have the code “quit out” like the “^” character behaves.

SRSRSD Requirement ID: 2.6.1.17

Institute a new Security Key for Group/Plan Edit (IBCN GROUP OR PLAN EDIT)

1. If Key exists the code will behave as the code currently does.
2. If the Key doesn't exist then:
 - a. If this is the correct Group/Plan then go to Policy prompt.

- b. If this isn't the correct Group/Plan then display an error message and "quit out" like the "^" character behaves.

3.5.2 Routines

Routine Name	NEW^IBCNBAA	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.4.1, 2.6.4.3, 2.6.4.4, and 2.6.4.5 Also 2.6.1.16 and 2.6.1.17	
Related Options	Add/Edit an Insurance Company Add/Edit a Group/Plan	
Related Routines	Routines "Called By"	Routines "Called"
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local	
Input Attribute Name and Definition		
Output Attribute Name and Definition		
Current Logic		
<pre> ; NEW(IBDESC) ; ask user if they want to add a new entry to the insurance files (36, 355.3, or 2.312) ; returns 1 if Yes create a new entry, 0 otherwise N DIR,X,Y,IBX S IBX=0 I IBDESC="INSURANCE COMPANY",\$D(^XUSEC("IB INSURANCE COMPANY ADD",DUZ)) W !,"Sorry, but you do not have the required privileges to add",!,"new Insurance Companies." D WAIT G NEWQ ; S DIR("?")="Enter Yes to create a new "_IBDESC_". Enter No to stop this process." S DIR("?",1)="Enter Yes to create a new "_IBDESC_" in the Insurance files for" S DIR("?",2)="this Buffer entry only if no existing "_IBDESC_" could be found" S DIR("?",3)="that matches this buffer entry.",DIR("?",4)=" W ! S DIR(0)="YO",DIR("A")="No "_IBDESC_" Selected, Add a New "_IBDESC_ D ^DIR I +Y=1 S IBX=1 NEWQ Q IBX ; </pre>		
Modified Logic (Changes are in bold)		

Routine Name	NEW^IBCNBAA
<p>Modifications to the NEW^IBCNBAA module of code, involve the instituting of 2 new security keys, "IB INSURANCE COMPANY EDIT" and "IB GROUP PLAN EDIT":</p> <pre> ; NEW(IBDESC) ; ask user if they want to add a new entry to the insurance files (36, 355.3, or 2.312) ; returns 1 if Yes create a new entry, 0 otherwise N DIR,X,Y,IBX S IBX=0 ;I IBDESC="INSURANCE COMPANY",\$D(^XUSEC("IB INSURANCE COMPANY ADD",DUZ)) W !,"Sorry, but you do not have the required privileges to add",!,"new Insurance Companies." D WAIT G NEWQ ; ;I IBDESC="INSURANCE COMPANY",\$D(^XUSEC("IB INSURANCE COMPANY EDIT",DUZ)) W !,"You must create an Insurance Company first.",!,"Please press RETURN to continue." D WAIT G NEWQ ;I IBDESC="GROUP/PLAN",\$D(^XUSEC("IB GROUP PLAN EDIT",DUZ)) W !,"You must create a Group Plan first.",!,"Please press RETURN to continue." D WAIT G NEWQ ; S DIR("?")="Enter Yes to create a new "_IBDESC_". Enter No to stop this process." S DIR("?",1)="Enter Yes to create a new "_IBDESC_" in the Insurance files for" S DIR("?",2)="this Buffer entry only if no existing "_IBDESC_" could be found" S DIR("?",3)="that matches this buffer entry.",DIR("?",4)="" W ! S DIR(0)="YO",DIR("A")="No "_IBDESC_" Selected, Add a New "_IBDESC D ^DIR I +Y=1 S IBX=1 NEWQ Q IBX ; </pre>	

Routine Name	AD^IBCNSM3	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.4.6	
Related Options	Patient Insurance Info View/Edit	
Related Routines	Routines "Called By"	Routines "Called"
		INS^IBCNSEH ADH^IBCNSM3 DUPCO^IBCNSOK1 SEL^IBCNSEH LK^IBCNSM31 NEW^IBCNSJ3 BEFORE^IBCNSEVT PAT^IBCNSEH PATPOL^IBCNSM32 POL^IBCNSEH AI^IBCNSP1
Data Dictionary (DD) References		

Routine Name	AD^IBCNSM3
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
<pre> ; AD ; -- Add new insurance policy N X,Y,DO,DD,DA,DR,DIC,DIE,DIK,DIR,DIRUT,IBCNSP,IBCPOL,IBQUIT,IBOK,IBCDFN,IBAD,IBGRP,IB ADPOL,IBCOVP,ANS,IBGNA,IBGNU S IBCNSEH=\$P(\$G(^IBE(350.9,1,4)),"^",1),IBQUIT=0,IBADPOL=1 D FULL^VALM1 S IBCOVP=\$P(\$G(^DPT(DFN,.31)),"^",11) I '\$D(^DPT(DFN,.312,0)) S ^DPT(DFN,.312,0)="^2.312PAI^^" ; D INS^IBCNSEH ; -- Select insurance company ; If one already exists for same co. ask are you sure you are ; adding a new one S DIR(0)="350.9,4.06" S DIR("A")="Select INSURANCE COMPANY",DIR("??")="^D ADH^IBCNSM3" S DIR(" ?")="Select the Insurance Company for the policy you are entering" D ^DIR K DIR S IBCNSP=+Y I Y<1 G ADQ I \$P(\$G(^DIC(36,IBCNSP,0)),"^",2)="N" W !,"This company does not reimburse. " I \$P(\$G(^DIC(36,IBCNSP,0)),"^",5) W !,*7,"Warning: Inactive Company" H 3 K IBCNSP G ADQ I \$\$DUPCO^IBCNSOK1(DFN,IBCNSP,"",1) H 3 ; ; -- see if can use existing policy D SEL^IBCNSEH S IBCPOL=\$\$LK^IBCNSM31(IBCNSP) I IBCPOL<1 D NEW^IBCNSJ3(IBCNSP,.IBCPOL) I IBCPOL<1 G ADQ ; ; -- file new patient policy S DIC("DR")=".18////" _IBCPOL_";1.09////1;1.05///NOW;1.06////" _DUZ K DD,DO S DA(1)=DFN,DIC="^DPT("_DFN_",.312," ,DIC(0)="L",X=IBCNSP D FILE^DICN K DIC S IBCDFN=+Y,IBNEW=1 I +Y<1 G ADQ D BEFORE^IBCNSEVT ; ; -- Edit patient policy data D PAT^IBCNSEH,PATPOL^IBCNSM32(IBCDFN) ; ; -- edit PLAN data if hold key I '\$D(^XUSEC("IB INSURANCE SUPERVISOR",DUZ)) G ADQ I '\$G(IBQUIT) D POL^IBCNSEH,EDPOL(IBCDFN) </pre>	

Routine Name	AD^IBCNSM3
<pre> I '\$G(IBNEW) D AI^IBCNSP1 G ADQ ; </pre>	
Modified Logic (Changes are in bold)	
<p>Modifications to the AD^IBCNSM3 module of code, involve the referencing of the new security keys, "IB GROUP PLAN EDIT":</p> <pre> ; AD ; -- Add new insurance policy N X,Y,DO,DD,DA,DR,DIC,DIE,DIK,DIR,DIRUT,IBCNSP,IBCPOL,IBQUIT,IBOK,IBCDFN,IBAD,IBGRP,IB ADPOL,IBCOVP,ANS,IBGNA,IBGNU S IBCNSEH=\$P(\$G(^IBE(350.9,1,4)),^",1),IBQUIT=0,IBADPOL=1 D FULL^VALM1 S IBCOVP=\$P(\$G(^DPT(DFN,.31)),^",11) I '\$D(^DPT(DFN,.312,0)) S ^DPT(DFN,.312,0)=""^2.312PAI^^" ; D INS^IBCNSEH ; -- Select insurance company ; If one already exists for same co. ask are you sure you are ; adding a new one S DIR(0)="350.9,4.06" S DIR("A")="Select INSURANCE COMPANY",DIR("??")=""^D ADH^IBCNSM3" S DIR(" ?")="Select the Insurance Company for the policy you are entering" D ^DIR K DIR S IBCNSP=+Y I Y<1 G ADQ I \$P(\$G(^DIC(36,IBCNSP,0)),^",2)="N" W !,"This company does not reimburse. " I \$P(\$G(^DIC(36,IBCNSP,0)),^",5) W !,*7,"Warning: Inactive Company" H 3 K IBCNSP G ADQ I \$\$DUPCO^IBCONSOK1(DFN,IBCNSP,"",1) H 3 ; ; -- see if can use existing policy D SEL^IBCNSEH S IBCPOL=\$\$LK^IBCNSM31(IBCNSP) I IBCPOL<1,\$D(^XUSEC("IB GROUP PLAN EDIT",DUZ)) W !,"Sorry, you are not authorized to create a new Insurance Plan",!,"Please press RETURN to continue." D WAIT G ADQ I IBCPOL<1 D NEW^IBCONS3(IBCNSP,IBCPOL) I IBCPOL<1 G ADQ ; ; -- file new patient policy S DIC("DR")="_.18////" _IBCPOL_";1.09////1;1.05///NOW;1.06////" _DUZ K DD,DO S DA(1)=DFN,DIC=""^DPT("_DFN_",.312,"",DIC(0)="L",X=IBCNSP D FILE^DICN K DIC S IBCDFN=+Y,IBNEW=1 I +Y<1 G ADQ D BEFORE^IBCNSEVT ; ; -- Edit patient policy data D PAT^IBCNSEH,PATPOL^IBCNSM32(IBCDFN) ; ; -- edit PLAN data if hold key I '\$D(^XUSEC("IB INSURANCE SUPERVISOR",DUZ)) G ADQ I '\$G(IBQUIT) D POL^IBCNSEH,EDPOL(IBCDFN) I '\$G(IBNEW) D AI^IBCNSP1 G ADQ </pre>	

Routine Name	AD^IBCNSM3
;	

3.5.3 Security Key

To satisfy requirements 2.6.4.2 and 2.6.4.6 the following new keys are needed:

Security Key Name	IB INSURANCE COMPANY EDIT			
Enhancement Category	<input checked="" type="checkbox"/> New <input type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change			
Related Options	Insurance Company Entry/Edit [IBCN INSURANCE CO EDIT]			
Related Routines	Routines “Called By”	Routines “Called”		
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output <input type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference			
Security Key Description	Key required to add insurance company to file #36. Used within the Process/Accept Insurance Buffer action.			
Subordinate Keys				
Mutually Exclusive Keys				
Granting Condition Logic				
Current Logic				
New – no current logic				
Modified Logic (Changes are in bold)				
Lock the option IBCN INSURANCE CO EDIT with the new security key IB INSURANCE COMPANY EDIT				
Hierarchical Precedence				

Security Key Name	IB GROUP PLAN EDIT			
Enhancement Category	<input checked="" type="checkbox"/> New <input type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change			
Related Options	Patient Insurance Info View/Edit [IBCN PATIENT INSURANCE]			
Related Routines	Routines “Called By”	Routines “Called”		
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output <input type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local Reference			
Security Key	Key required to add insurance group/plan to file #355.3. Used within the			

Description	Process/Accept Insurance Buffer action and Patient Insurance Info View/Edit (when adding a policy).
Subordinate Keys	
Mutually Exclusive Keys	
Granting Condition Logic	
Current Logic	
New – no current logic	
Modified Logic (Changes are in bold)	
Add the security key to restrict users from adding/editing the Group Plan for a patient while using the “Patient Insurance Info View/Edit” [IBCN PATIENT INSURANCE] option. Modifications needed to do this are detailed above AR^IBCNSM3 (section 3.5.2 of this document).	
Hierarchical Precedence	

3.6 System Feature: Enhancements to Patient’s Eligibility Benefits

This section covers the design for the requirements listed in Section 2.6.5 of the RSD.

3.6.1 Functional Requirements:

Table 5: Enhancements to Patient’s Eligibility Benefits

SRS RSD REQ ID	REQ Title	Comments / Notes
2.6.5.1	Eligibility Benefits – Update the Eligibility Benefit Information Accessed via the ‘Process Insurance Buffer’ option	
2.6.5.2	Eligibility Benefits – Update the Eligibility Benefit Information Accessed via the ‘TPJI’ Option	
2.6.5.3	Eligibility Benefits – Update the Eligibility Benefit Information Accessed via the ‘Patient Insurance Info View/Edit’ Option	
2.6.5.4	Eligibility Benefits – Include the Eligibility Benefit Information on the eIV Response Report	

<u>SRSRSD</u> REQ ID	REQ Title	Comments / Notes
2.6.5.5	Eligibility Benefits – Store the Service Date on the Patient's Policy Record	This is the eligibility/service date that was received on the response (X12 271 message). If that is not present then this is the eligibility date that was sent on the inquiry (X12 270 message). When storing this on the patient's record refer to this as the 'Requested Service Date' to help make it clear what it is.
2.6.5.6	Eligibility Benefits – Store the Service Type on the Patient's Policy Record	This is the service type that was sent on the inquiry (X12 270 message). When storing this on the patient's record refer to this as the 'Requested Service Type' to help make it clear what it is.
2.6.5.7	Eligibility Benefits – Display the Service Date of the Response	This is the requested eligibility/service date that was stored on the patient's record.
2.6.5.8	Eligibility Benefits – Display the Service Type of the Response	This is the requested service type that was stored on the patient's record.
2.6.5.9	Eligibility Benefits – Eligibility Benefit (1st Priority Sort Order): Insurance Status	CBO asked that we use the phrase "Coverage Status" when we display the information to the user.
2.6.5.10	Eligibility Benefits - Eligibility Benefit (2nd Priority Sort Order): Insurance Type	
2.6.5.11	Eligibility Benefits – Eligibility Benefit (3rd Priority Sort Order): Coordination of Benefits (COB) (REMOVED)	Removed on 8/8/13 and is referenced in the <u>SRSRSD</u> version 1.2
2.6.5.12	Eligibility Benefits - Eligibility Benefit (4th Priority Sort Order): Indication of Other Insurance	

3.6.2 Routines

Routine Name	INIT^IBCNES
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Requirement Traceability Matrix	2.6.5.1, 2.6.5.2, 2.6.5.3, 2.6.5.4, 2.6.5.9, 2.6.5.10, 2.6.5.12
Related Options	Process Insurance Buffer

Routine Name	INIT^IBCNE5	
	Third Party Joint Inquiry Patient Insurance Infor View/Edit eIV Response Report	
Related Routines	Routines "Called By"	Routines "Called"
	^IBCNCBD ^IBCNS3	RPDM^IBCNE53 FO^IBCNEUT1 SET^IBCNE51 EB^IBCNE51 CMPI^IBCNE51 HCSD^IBCNE51 NTE^IBCNE51 BRE^IBCNE51
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local	
Input Attribute Name and Definition		
Output Attribute Name and Definition		
Current Logic		
<pre> ; INIT(IBVF,IBVIENS,IBVEBFLG,IBVV,IBVSUB) ; List Entry ; ; IBVF = file# 2.322 or 365.02 (required) ; IBVIENS = std IENS list of internal entry numbers - NOT including any EB iens (required) ; IBVEBFLG = flag indicating which EB records to pull ; "A" - all of them ; "L" - only the last one (default) ; "F" - only the first one ; "M" - multiple, pass IBEBFLG by reference and include the IB iens in ; an array as follows: ; IBVEBFLG="M" ; IBVEBFLG(3)=" " ; IBVEBFLG(5)=" " ; IBVV = Video attributes flag ; 1 = reverse video (default) ; 2 = bold ; 3 = underline ; IBVSUB = literal subscript to use in the display scratch global ; ; </pre>		

Routine Name	INIT^IBCNE5
<pre> NEW IBVDA,GLO,IBVLIST,IEN,IBVEBIEN,IBVEBTOT,IBVEBCNT ; S IBVSUB=\$G(IBVSUB) I IBVSUB="" S IBVSUB="EB ELIG/BEN" K ^TMP(IBVSUB,\$J) I \$D(VALMEVL) D CLEAN^VALM10,KILL^VALM10() ; D DA^DILF(IBVIENS,.IBVDA) ; build the IBVDA array for the iens I '\$D(IBVDA) D NODATA G INITX ; I \$D(VALMEVL),\$G(IBVV) S IBVV=1 ; default reverse video for ListMan I '\$D(VALMEVL) S IBVV="" ; no video attributes for non-ListMan ; D RPDMA^IBCNE53(\$S(IBVF=365.02:365,1:2.312),.IBVDA,IBVV,IBVSUB) ; IB*2*497 display group level eligibility information ; I IBVF=2.322 S GLO=\$NA(^DPT(+\$G(IBVDA(1)),.312,\$G(IBVDA),6)) ; pt. insurance I IBVF=365.02 S GLO=\$NA(^BCN(365,\$G(IBVDA),2)) ; response file I \$G(GLO)="" D NODATA G INITX ; S IBVEBFLG=\$G(IBVEBFLG,"L") K IBVLIST I IBVEBFLG="L" S IEN=+\$O(@GLO@(" "),-1) I IEN S IBVLIST(IEN)="" ; last EB ien on file I IBVEBFLG="F" S IEN=+\$O(@GLO@(0)) I IEN S IBVLIST(IEN)="" ; first EB ien on file I IBVEBFLG="A" S IEN=0 F S IEN=\$O(@GLO@(IEN)) Q:'IEN S IBVLIST(IEN)="" ; all EB iens on file I IBVEBFLG="M" S IEN=0 F S IEN=\$O(IBVEBFLG(IEN)) Q:'IEN I \$D(@GLO@(IEN)) S IBVLIST(IEN)="" ; multiple ; I '\$D(IBVLIST) D NODATA G INITX ; ; count them S IEN=0 F IBVEBTOT=0:1 S IEN=\$O(IBVLIST(IEN)) Q:'IEN I 'IBVEBTOT D NODATA G INITX ; S (IBVEBIEN,IBVEBCNT)=0 F S IBVEBIEN=\$O(IBVLIST(IBVEBIEN)) Q:'IBVEBIEN D . S IBVEBCNT=IBVEBCNT+1 . N TXVIENS . ; . ; if there is more than 1 EB group, then display a header line for separation . I IBVEBTOT>1 D .. N DSP,LN,IBZ .. S DSP=\$NA(^TMP(IBVSUB,\$J,"DISP")) .. S LN=+\$O(@DSP@(""),-1) .. S IBZ="eIV Eligibility/Benefit Data Group# "_IBVEBCNT_" of "_IBVEBTOT .. S IBZ=\$\$FO^IBCNEUT1(\$J(" ",20)_IBZ,80) .. S LN=LN+1 D SET^IBCNE51(LN,1,IBZ,,IBVV) .. S LN=LN+1 D SET^IBCNE51(LN) .. Q . ; . ; add this EB ien to the list of iens . S TXVIENS=IBVEBIEN_" "_IBVIENS . ; </pre>	

Routine Name	INIT^IBCNE\$
	<pre> ; ; call the screen sections to build the display . D EB^IBCNE\$1(IBVF, TXVIENS, IBVV, IBVSUB) . D CMPI^IBCNE\$1(IBVF, TXVIENS, IBVV, IBVSUB) . D HCSD^IBCNE\$1(IBVF, TXVIENS, IBVV, IBVSUB) . D NTE^IBCNE\$1(IBVF, TXVIENS, IBVV, IBVSUB) . D BRE^IBCNE\$1(IBVF, TXVIENS, IBVV, IBVSUB) ; ; . Q ; ; S VALMCNT=\$O(^TMP(IBVSUB,\$J,"DISP"," "),-1) ; INITX ; Q ; ; </pre>
	Modified Logic (Changes are in bold)
	<pre> ; INIT(IBVF,IBVIENS,IBVEBFLG,IBVV,IBVSUB) ; List Entry ; ; ; IBVF = file# 2.322 or 365.02 (required) ; IBVIENS = std IENS list of internal entry numbers - NOT including any EB iens (required) ; IBVEBFLG = flag indicating which EB records to pull ; "A" - all of them ; "L" - only the last one (default) ; "F" - only the first one ; "M" - multiple, pass IBEBFLG by reference and include the IB iens in ; an array as follows: ; IBVEBFLG="M" ; IBVEBFLG(3)=" ; IBVEBFLG(5)=" ; IBVV = Video attributes flag ; 1 = reverse video (default) ; 2 = bold ; 3 = underline ; IBVSUB = literal subscript to use in the display scratch global ; NEW IBVDA,GLO,IBVLIST,IEN,IBVEBIEN,IBVEBTOT,IBVEBCNT ; S IBVSUB=\$G(IBVSUB) I IBVSUB="" S IBVSUB="EB ELIG/BEN" K ^TMP(IBVSUB,\$J) I \$D(VALMEVL) D CLEAN^VALM10,KILL^VALM10() ; D DA^DILF(IBVIENS,.IBVDA) ; build the IBVDA array for the iens I '\$D(IBVDA) D NODATA G INITX ; ; I \$D(VALMEVL),\$G(IBVV) S IBVV=1 ; default reverse video for ListMan I '\$D(VALMEVL) S IBVV="" ; no video attributes for non-ListMan ; ; </pre>

Routine Name	INIT^IBCNE3
D RPDM^IBCNE3(\$S(IBVF=365.02:365,1:2.312),,IBVDA,IBVV,IBVSUB) ; IB*2*497 display group level eligibility information ; I IBVF=2.322 S GLO=\$NA(^DPT(+ \$G(IBVDA(1)),.312,+ \$G(IBVDA),6)) ; pt. insurance I IBVF=365.02 S GLO=\$NA(^IBCN(365,+ \$G(IBVDA),2)) ; response file I \$G(GLO)=" " D NODATA G INITX ; S IBVEBFLG=\$G(IBVEBFLG,"L") K IBVLIST I IBVEBFLG="L" S IEN=+\$O(@GLO@(" "),-1) I IEN S IBVLIST(IEN)=" " ; last EB ien on file I IBVEBFLG="F" S IEN=+\$O(@GLO@(0)) I IEN S IBVLIST(IEN)=" " ; first EB ien on file I IBVEBFLG="A" S IEN=0 F S IEN=\$O(@GLO@(IEN)) Q:'IEN S IBVLIST(IEN)=" " ; all EB iens on file I IBVEBFLG="M" S IEN=0 F S IEN=\$O(IBVEBFLG(IEN)) Q:'IEN I \$D(@GLO@(IEN)) S IBVLIST(IEN)=" " ; multiple ; I '\$D(IBVLIST) D NODATA G INITX ; ; count them S IEN=0 F IBVEBTOT=0:1 S IEN=\$O(IBVLIST(IEN)) Q:'IEN I 'IBVEBTOT D NODATA G INITX ; ; DO SUMMARY ; S (IBVEBIEN,IBVEBCNT)=0 F S IBVEBIEN=\$O(IBVLIST(IBVEBIEN)) Q:'IBVEBIEN D . S IBVEBCNT=IBVEBCNT+1 . N TXVIENS . ; . ; if there is more than 1 EB group, then display a header line for separation . I IBVEBTOT>1 D .. N DSP, LN, IBZ .. S DSP=\$NA(^TMP(IBVSUB,\$J,"DISP")) .. S LN=+\$O(@DSP@(" "),-1) .. S IBZ="eIV Eligibility/Benefit Data Group# "_IBVEBCNT_" of "_IBVEBTOT .. S IBZ=\$\$FO^IBCNEUT1(\$J(" " ,20)_IBZ,80) .. S LN=LN+1 D SET^IBCNE31(LN,1,IBZ,,IBVV) .. S LN=LN+1 D SET^IBCNE31(LN) .. Q . ; . ; add this EB ien to the list of iens . S TXVIENS=IBVEBIEN_"_"_IBVIENS . ; . ; call the screen sections to build the display . D EB^IBCNE31(IBVF,TXVIENS,IBVV,IBVSUB) . D CMPI^IBCNE31(IBVF,TXVIENS,IBVV,IBVSUB)	

Routine Name	INIT^IBCNE5
<pre> . D HCSD^IBCNE51(IBVF, TXVIENS, IBVV, IBVSUB) . D NTE^IBCNE51(IBVF, TXVIENS, IBVV, IBVSUB) . D BRE^IBCNE51(IBVF, TXVIENS, IBVV, IBVSUB) . ; . Q ; S VALMCNT=\$O(^TMP(IBVSUB,\$J,"DISP",""),-1) ; INITX ; Q ; SUMMARY; (new tag) list key elements from the Eligibility Benefit information N DSP, LN, IBZ, OTHRINS, STATUS, TYPE ; S DSP=\$NA(^TMP(IBVSUB,\$J,"DISP")) S LN=+ \$O(@DSP@(""),-1) S IBZ="Summary of eV Eligibility/Benefit Data" S IBZ=\$\$FO^IBCNEUT1(\$J(" ",20)_IBZ,80) S LN=LN+1 D SET^IBCNE51(LN,1,IBZ,,IBVV) S LN=LN+1 D SET^IBCNE51(LN) ; ; loop through all EBs and identify the following elements, if any of the EBs contradict another ; EB then the field in question will be set to unknown or ambiguous as stated below. ; ; coverage status is field ELIGIBILITY/BENEFIT INFO (#365.011,.02) . Only look at values 1, 6, ; and V to determine the coverage status. "1" = ACTIVE, "6" = INACTIVE, "V" = Ambiguous. ; set STATUS = either "Active", "Inactive", or "Ambiguous" based on the EBs ; ; INSURANCE TYPE is (#365.014,.05). If there is a conflict in the values found, then set ; TYPE="Unknown" otherwise TYPE= the external value of INSURANCE TYPE ; S LN=LN+1 D SET^IBCNE51(LN,1," Coverage Status: ", STATUS) S LN=LN+1 D SET^IBCNE51(LN,1," Insurance Type: ", TYPE) S OTHRINS=0 ; If the response is from Medicare - loop thru all EBs and if any of them have an eligibility code ; (#.02) ="R" then set variable OTHRINS=1 I OTHRINS S LN=LN+1 D SET^IBCNE51(LN,1,"Other insurance was potentially found") Q </pre>	
Routine Name	GETDATA^IBCNERPE

Routine Name	GETDATA^IBCNERPE	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.5.4	
Related Options	eIV Response Report	
Related Routines	Routines "Called By"	Routines "Called"
		X12^IBCNERP2
Data Dictionary (DD) References		
Related Protocols		
Related Integration Control Registrations (ICRs)		
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local	
Input Attribute Name and Definition		
Output Attribute Name and Definition		
Current Logic		
<pre> ; GETDATA(IEN,RPTDATA) ; Retrieve response data ; Init N %,CNPTR,CT,DIW,DIWI,DIWT,DIWTC,DIWX,DN,EACT,ELOC,ESRC,ETXT,DQUAL,DTYPE,FUTDT,I ENS,II,LOOP,NODE0,PC,TQIEN,Z ; ; Insured Info from eIV Response #365 S RPTDATA(0)=\$G(^IBC(365,IEN,0)),TQIEN=\$P(RPTDATA(0),U,5) ; Trans dates to ext format S \$P(RPTDATA(0),U,7)=\$\$FMTE^XLFD(\$P(RPTDATA(0),U,7)\1,"5Z") S RPTDATA(1)=\$G(^IBC(365,IEN,1)) S ^CLT(0)=RPTDATA(1) ; Trans ext values for SET of CODES values S IENS=IEN_" " S \$P(RPTDATA(1),U,8)=\$\$GET1^DIQ(365,IENS,1.08,"E") ; Whose Ins S \$P(RPTDATA(1),U,13)=\$\$GET1^DIQ(365,IENS,1.13,"E") ; COB S RPTDATA(8)=\$\$GET1^DIQ(365,IENS,8.01,"E") ; Pt Rel to Sub ; if pt. rel is empty, try to get value from the old field 365/1.09 I RPTDATA(8)="" S RPTDATA(8)=\$\$GET1^DIQ(365,IENS,1.09,"E") ; Trans err actions/codes to ext S \$P(RPTDATA(1),U,14)=\$\$X12^IBCNERP2(365.017,\$P(RPTDATA(1),U,14)) S \$P(RPTDATA(1),U,15)=\$\$X12^IBCNERP2(365.018,\$P(RPTDATA(1),U,15)) ; Trans dates to ext format - check format F PC=2,9:1:12,16,17,19 S \$P(RPTDATA(1),U,PC)=\$\$FMTE^XLFD(\$P(RPTDATA(1),U,PC),"5Z") ; ; Loop thru mult Contact segs S CT=0 </pre>		

Routine Name	GETDATA^IBCNERPE
<pre> F S CT=\$O(^IBC(365,IEN,3,CT)) Q:'CT D .S RPTDATA(3,CT)=\$G(^IBC(365,IEN,3,CT,0)) .; Obtain the various Communication Text fields .F II=1:1:3 S RPTDATA(3,CT,II)=\$G(^IBC(365,IEN,3,CT,II)) .; Disp. blank if NOT SPECIFIED .I \$P(RPTDATA(3,CT),U)="NOT SPECIFIED" S \$P(RPTDATA(3,CT),U)=" .; Comm Qual #1-3 .F II=1:1:3 D ..S CNPTR=\$\$X12^IBCNERP2(365.021,\$P(RPTDATA(3,CT),U,II*2)) ...;;I CNPTR="" S \$P(RPTDATA(3,CT),U,II*2)=CNPTR_: "_\$P(RPTDATA(3,CT),U,II*2+1),\$P(RPTDATA(3,CT),U,II*2+1)=" ..I CNPTR="" S RPTDATA(3,CT,II)=CNPTR_: "_\$G(RPTDATA(3,CT,II)) ; ; Subscriber level dates (ZTP segments) S CT=0 F S CT=\$O(^IBC(365,IEN,7,CT)) Q:'CT D .S NODE0=\$G(^IBC(365,IEN,7,CT,0)) .S DQUAL=\$P(NODE0,U,3) I 'DQUAL Q .S LOOP=\$\$GET1^DIQ(365.027,\$P(NODE0,U,4)_"",",.01) .S DTYPE=\$S(LOOP["C": "S", LOOP["D": "P", 1: "O") .S RPTDATA(7,DTYPE,CT)=\$\$X12^IBCNERP2(365.026,DQUAL)_U_\$P(NODE0,U,2) .Q ; ; Reject reasons S CT=0 F S CT=\$O(^IBC(365,IEN,6,CT)) Q:'CT D .S NODE0=\$G(^IBC(365,IEN,6,CT,0)) I ' \$P(NODE0,U,3) Q .S ETXT=\$\$X12^IBCNERP2(365.017,\$P(NODE0,U,3)) .S ELOC=\$P(NODE0,U,2) S:ELOC="" ELOC="N/A" .S EACT=\$\$X12^IBCNERP2(365.018,\$P(NODE0,U,4)) S:EACT="" EACT="N/A" .S LOOP=\$\$X12^IBCNERP2(365.027,\$P(NODE0,U,5)) S:LOOP="" LOOP="N/A" .S ESRC=\$P(NODE0,U,6) S:ESRC="" ESRC="N/A" .;IB*2*497 modify existing line below to retrieve external value of ERROR CODE .;and build as part of the composite string at RPTDATA(6,CT). .S RPTDATA(6,CT)=ELOC_U_\$\$GET1^DIQ(365.017,\$P(NODE0,U,3)_"",",.01)_U_ETXT_U_EACT_U_L OOP_U_ESRC .; IB*2*497 retrieve additional messages .S Z=0 F S Z=\$O(^IBC(365,IEN,6,CT,1,Z)) Q:'Z S RPTDATA(6,CT,"AMSG",Z)=\$P(\$G(^IBC(365,IEN,6,CT,1,Z,0)),U) .Q ; ; Subscriber Data S RPTDATA(13)=\$G(^IBC(365,IEN,13)) ; ; Group Data S RPTDATA(14)=\$G(^IBC(365,IEN,14)) ; FUTDT I TQIEN D ; If there is a future date, display it .S FUTDT=\$P(\$G(^IBC(365.1,TQIEN,0)),U,9) Q:FUTDT="" .S II=\$O(RPTDATA(5,""),-1)+1 .S RPTDATA(5,II)=" ",II=II+1 .S RPTDATA(5,II)="Inquiry will be automatically resubmitted on " _\$\$FMTE^XLFD(FUTDT,"5Z")_" " ; GETDATX ; GETDATA exit point Q </pre>	

Routine Name	GETDATA^IBCNERPE
<pre> ; ; This tag is only called from IBCNERP3 ; DATA(DISPDATA) ; Build disp lines N LCT,CT,SEGCT,ITEM,CT2,NTCT,CNCT,ERCT,RPTDATA,DCT,DTYPE ; Merge into local array M RPTDATA=^TMP(\$J,RTN,SORT1,SORT2,CNT) ; Build S LCT=1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,13.01),17,"R")_P(RPTDATA(13),U,1) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,13.02),17,"R")_P(RPTDATA(13),U,2) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.02),17,"R")_P(RPTDATA(1),U,2) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.03),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(1),U,3),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.04),22,"R")_P(RPTDATA(1),U,4) S LCT=LCT+1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,14.01),17,"R")_P(RPTDATA(14),U,1) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,14.02),17,"R")_P(RPTDATA(14),U,2) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.08),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(1),U,8),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,8.01),22,"R")_RPTDATA(8) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.18),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(1),U,18),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.13),22,"R")_P(RPTDATA(1),U,13) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.1),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(1),U,10),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.16),22,"R")_P(RPTDATA(1),U,16) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.11),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(1),U,11),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.17),22,"R")_P(RPTDATA(1),U,17) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.12),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(1),U,12),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.19),22,"R")_P(RPTDATA(1),U,19) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,.07),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(0),U,7),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,.09),22,"R")_P(RPTDATA(0),U,9) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.2),17,"R")_\$\$FO^IBCNEUT1(P(RPTDATA(1),U,20),20) ; ; Dates F DTYPE="S","P","O" D .I '\$D(RPTDATA(7,DTYPE)) Q .S LCT=LCT+1,DISPDATA(LCT)=" " .S LCT=LCT+1,DISPDATA(LCT)=\$\$S(DTYPE="S": "Subscriber",DTYPE="P": "Patient",1: "Other")_ " Dates:" .S LCT=LCT+1,DISPDATA(LCT)=" " .S DCT=" " F S DCT=\$O(RPTDATA(7,DTYPE,DCT)) Q: DCT=" " D ..S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(P(RPTDATA(7,DTYPE,DCT),U)_ ":",40)_P(RPTDATA(7,DTYPE,DCT),U,2) ..Q .Q ; </pre>	

Routine Name	GETDATA^IBCNERPE
<pre> ; Contacts CONT ; N TEXT S CNCT=+\$O(RPTDATA(3,""),-1) I 'CNCT G ERR S DISPDATA(LCT)="",LCT=LCT+1,DISPDATA(LCT)="CONTACT INFORMATION:",LCT=LCT+1 ; Build F CT=1:1:CNCT D . S DISPDATA(LCT)="",LCT=LCT+1,DISPDATA(LCT)=" " . S SEGCT=\$O(RPTDATA(3,CT,""),-1) . S (DISPDATA(LCT),TEXT)=" " . I \$L(\$P(RPTDATA(3,CT),U,1)) S TEXT=\$P(RPTDATA(3,CT),U,1) . F CT=1:1:SEGCT S ITEM=\$G(RPTDATA(3,CT,CT2)) D .. Q:\$L(ITEM) .. S TEXT=\$S(\$L(TEXT):" "_TEXT_" ",1:" ")_ITEM .. F D Q:\$L(TEXT) ... S DISPDATA(LCT)=\$E(TEXT,1,74) ... S LCT=LCT+1 ... I \$L(TEXT)>74 S TEXT=\$E(TEXT,75,\$L(TEXT)) Q ... S TEXT="" ... Q .. Q ; Err Info ERR S ERCT=+\$O(RPTDATA(6,""),-1) I 'ERCT G DATA S DISPDATA(LCT)="",LCT=LCT+1 S DISPDATA(LCT)="ERROR INFORMATION:",LCT=LCT+1 S DISPDATA(LCT)=" " F CT=1:1:ERCT D . S LCT=LCT+1,DISPDATA(LCT)="Reject Reason Code: " _\$P(RPTDATA(6,CT),U,2) ; ib*2*497 . S LCT=LCT+1,DISPDATA(LCT)="Reject Reason Text: " _\$P(RPTDATA(6,CT),U,3) ; ib*2*497 . S LCT=LCT+1,DISPDATA(LCT)="Action Code: " _\$P(RPTDATA(6,CT),U,3) . S LCT=LCT+1,DISPDATA(LCT)="HIPAA Loop: " _\$P(RPTDATA(6,CT),U,4) . S LCT=LCT+1,DISPDATA(LCT)="HL7 Location: " _\$P(RPTDATA(6,CT),U) . S LCT=LCT+1,DISPDATA(LCT)="Error Source: " _\$P(RPTDATA(6,CT),U,5) . I \$D(RPTDATA(6,CT,"AMSG")) D .. S LCT=LCT+1,DISPDATA(LCT)=" " .. S LCT=LCT+1,DISPDATA(LCT)="Additional Messages:" .. S LCT=LCT+1,DISPDATA(LCT)=" " .. S Z=0 F S Z=\$O(RPTDATA(6,CT,"AMSG",Z)) Q:'Z S LCT=LCT+1,DISPDATA(LCT)=RPTDATA(6,CT,"AMSG",Z) .. Q . S LCT=LCT+1,DISPDATA(LCT)=" " . Q ; DATA ; ; Disp Future Date and Misc. Comments I \$O(RPTDATA(5,0))="" D . F CT=1:1:+\$O(RPTDATA(5,""),-1) D .. S DISPDATA(LCT)=" " _\$FO^IBCNEUT1(" ",7,"R")_\$G(RPTDATA(5,CT)),LCT=LCT+1 ; ; Q ; </pre>	

Routine Name	GETDATA^IBCNERPE
Modified Logic (Changes are in bold)	
<pre> ; GETDATA(IEN,RPTDATA) ; Retrieve response data ; Init N %,CNPTR,CT,DIW,DIWI,DIWT,DIWTC,DIWX,DN,EACT,ELOC,ESRC,ETXT,DQUAL,DTYPE,FUTDT,I ENS,II,LOOP,NODE0,PC,TQIEN,Z ; ; Insured Info from eIV Response #365 S RPTDATA(0)=\$G(^IBC(365,IEN,0)),TQIEN=\$P(RPTDATA(0),U,5) ; Trans dates to ext format S \$P(RPTDATA(0),U,7)=\$FMTE^XLFD(\$P(RPTDATA(0),U,7)\1,"5Z") S RPTDATA(1)=\$G(^IBC(365,IEN,1)) S ^CLT(0)=RPTDATA(1) ; Trans ext values for SET of CODES values S IENS=IEN_"", S \$P(RPTDATA(1),U,8)=\$\$GET1^DIQ(365,IENS,1.08,"E") ; Whose Ins S \$P(RPTDATA(1),U,13)=\$\$GET1^DIQ(365,IENS,1.13,"E") ; COB S RPTDATA(8)=\$\$GET1^DIQ(365,IENS,8.01,"E") ; Pt Rel to Sub ; if pt. rel is empty, try to get value from the old field 365/1.09 I RPTDATA(8)="" S RPTDATA(8)=\$\$GET1^DIQ(365,IENS,1.09,"E") ; Trans err actions/codes to ext S \$P(RPTDATA(1),U,14)=\$\$X12^IBCNERP2(365.017,\$P(RPTDATA(1),U,14)) S \$P(RPTDATA(1),U,15)=\$\$X12^IBCNERP2(365.018,\$P(RPTDATA(1),U,15)) ; Trans dates to ext format - check format F PC=2,9:1:12,16,17,19 S \$P(RPTDATA(1),U,PC)=\$FMTE^XLFD(\$P(RPTDATA(1),U,PC),"5Z") ; ; Loop thru mult Contact segs S CT=0 F S CT=\$O(^IBC(365,IEN,3,CT)) Q:'CT D .S RPTDATA(3,CT)=\$G(^IBC(365,IEN,3,CT,0)) .; Obtain the various Communication Text fields .F II=1:1:3 S RPTDATA(3,CT,II)=\$G(^IBC(365,IEN,3,CT,II)) .; Disp. blank if NOT SPECIFIED .I \$P(RPTDATA(3,CT),U)="NOT SPECIFIED" S \$P(RPTDATA(3,CT),U)="" .; Comm Qual #1-3 .F II=1:1:3 D ..S CNPTR=\$\$X12^IBCNERP2(365.021,\$P(RPTDATA(3,CT),U,II*2)) ...;;I CNPTR="" S \$P(RPTDATA(3,CT),U,II*2)=CNPTR_": "_\$P(RPTDATA(3,CT),U,II*2+1),\$P(RPTDATA(3,CT),U,II*2+1)="" ..I CNPTR="" S RPTDATA(3,CT,II)=CNPTR_": "_\$G(RPTDATA(3,CT,II)) ; ; Subscriber level dates (ZTP segments) S CT=0 F S CT=\$O(^IBC(365,IEN,7,CT)) Q:'CT D .S NODE0=\$G(^IBC(365,IEN,7,CT,0)) .S DQUAL=\$P(NODE0,U,3) I 'DQUAL Q .S LOOP=\$\$GET1^DIQ(365.027,\$P(NODE0,U,4)_",",.01) .S DTYPE=\$\$S(Loop["C": "S",Loop["D": "P",1: "O") .S RPTDATA(7,DTYPE,CT)=\$\$X12^IBCNERP2(365.026,DQUAL)_U_\$P(NODE0,U,2) .Q ; ; Reject reasons S CT=0 F S CT=\$O(^IBC(365,IEN,6,CT)) Q:'CT D .S NODE0=\$G(^IBC(365,IEN,6,CT,0)) I '\$P(NODE0,U,3) Q </pre>	

Routine Name	GETDATA^IBCNERPE
<pre> .S ETXT=\$\$X12^IBCNERP2(365.017,\$P(NODE0,U,3)) .S ELOC=\$P(NODE0,U,2) S:ELOC="" ELOC="N/A" .S EACT=\$\$X12^IBCNERP2(365.018,\$P(NODE0,U,4)) S:EACT="" EACT="N/A" .S LOOP=\$\$X12^IBCNERP2(365.027,\$P(NODE0,U,5)) S:LOOP="" LOOP="N/A" .S ESRC=\$P(NODE0,U,6) S:ESRC="" ESRC="N/A" .;IB*2*497 modify existing line below to retrieve external value of ERROR CODE .;and build as part of the composite string at RPTDATA(6,CT). .S RPTDATA(6,CT)=ELOC_U_\$\$GET1^DIQ(365.017,\$P(NODE0,U,3),"",.01)_U_ETXT_U_EACT_U_L OOP_U_ESRC .; IB*2*497 retrieve additional messages .S Z=0 F S Z=\$O(^IBCN(365,IEN,6,CT,1,Z)) Q:'Z S RPTDATA(6,CT,"AMSG",Z)=\$P(\$G(^IBCN(365,IEN,6,CT,1,Z,0)),U) .Q ; ; ; Subscriber Data S RPTDATA(13)=\$G(^IBCN(365,IEN,13)) ; ; Group Data S RPTDATA(14)=\$G(^IBCN(365,IEN,14)) ; FUTDT I TQIEN D ; If there is a future date, display it .S FUTDT=\$P(\$G(^IBCN(365.1,TQIEN,0)),U,9) Q:FUTDT="" .S II=\$O(RPTDATA(5,""),-1)+1 .S RPTDATA(5,II)=" ",II=II+1 .S RPTDATA(5,II)="Inquiry will be automatically resubmitted on "_\$\$FMTE^XLFD(FUTDT,"5Z")_" ; GETDATX ; GETDATA exit point Q ; ; ; This tag is only called from IBCNERP3 ; DATA(DISPDATA) ; Build disp lines N LCT,CT,SEGCT,ITEM,CT2,NTCT,CNCT,ERCT,RPTDATA,DCT,DTYPE ; Merge into local array M RPTDATA=^TMP(\$J,RTN,SORT1,SORT2,CNT) ; Build S LCT=1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,13.01),17,"R")_P(RPTDATA(13),U,1) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,13.02),17,"R")_P(RPTDATA(13),U,2) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.02),17,"R")_P(RPTDATA(1),U,2) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.03),17,"R")_\$\$FO^IBCNE UT1(P(RPTDATA(1),U,3),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.04),22,"R")_P(RPTDATA(1),U,4) S LCT=LCT+1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,14.01),17,"R")_P(RPTDATA(14),U,1) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,ITEM=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,14.02),17,"R")_P(RPTDATA(14),U,2) D WRAPIT(ITEM,.LCT,.DISPDATA,74,17) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.08),17,"R")_\$\$FO^IBCNE UT1(P(RPTDATA(1),U,8),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,8.01),22,"R")_RPTDATA(8) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.18),17,"R")_\$\$FO^IBCNE UT1(P(RPTDATA(1),U,18),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.13),22,"R")_P(RPTDATA(1 </pre>	

Routine Name	GETDATA^IBCNERPE
<pre>),U,13) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.1),17,"R")_\$\$FO^IBCNEU T1(\$P(RPTDATA(1),U,10),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.16),22,"R")_ \$P(RPTDATA(1), U,16) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.11),17,"R")_\$\$FO^IBCNE UT1(\$P(RPTDATA(1),U,11),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.17),22,"R")_ \$P(RPTDATA(1),U,17) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.12),17,"R")_\$\$FO^IBCNE UT1(\$P(RPTDATA(1),U,12),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.19),22,"R")_ \$P(RPTDATA(1),U,19) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,.07),17,"R")_\$\$FO^IBCNEU T1(\$P(RPTDATA(0),U,7),20)_\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,.09),22,"R")_ \$P(RPTDATA(0),U, 9) S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$\$LBL^IBCNERP2(365,1.2),17,"R")_\$\$FO^IBCNEU T1(\$P(RPTDATA(1),U,20),20) ; ; Dates F DTYPE="S","P","O" D .I '\$D(RPTDATA(7,DTYPE)) Q .S LCT=LCT+1,DISPDATA(LCT)=" " .S LCT=LCT+1,DISPDATA(LCT)=\$S(DTYPE="S": "Subscriber",DTYPE="P": "Patient",1: "Other")_ " Dates:" .S LCT=LCT+1,DISPDATA(LCT)=" " .S DCT=" " F S DCT=\$O(RPTDATA(7,DTYPE,DCT)) Q: DCT=" " D ..S LCT=LCT+1,DISPDATA(LCT)=\$\$FO^IBCNEUT1(\$P(RPTDATA(7,DTYPE,DCT),U)_": ",40)_ \$P(RPTDATA(7,DTYPE,DCT),U,2) ..Q .Q ; ; ; Contacts CONT ; N TEXT S CNCT=+\$O(RPTDATA(3,""),-1) I 'CNCT G ERR S DISPDATA(LCT)="",LCT=LCT+1,DISPDATA(LCT)="CONTACT INFORMATION:",LCT=LCT+1 ; Build F CT=1:1:CNCT D . S DISPDATA(LCT)="",LCT=LCT+1,DISPDATA(LCT)=" " . S SEGCT=\$O(RPTDATA(3,CT,""),-1) . S (DISPDATA(LCT),TEXT)=" " . I \$L(\$P(RPTDATA(3,CT),U,1)) S TEXT=\$P(RPTDATA(3,CT),U,1) . F CT2=1:1:SEGCT S ITEM=\$G(RPTDATA(3,CT,CT2)) D .. Q: '\$L(ITEM) .. S TEXT=\$S(\$L(TEXT): " _TEXT_ ",1: " ")_ITEM .. F D Q: '\$L(TEXT) ... S DISPDATA(LCT)=\$E(TEXT,1,74) ... S LCT=LCT+1 ... I \$L(TEXT)>74 S TEXT=\$E(TEXT,75,\$L(TEXT)) Q ... S TEXT=" " ... Q .. Q ; Err Info ERR S ERCT=+\$O(RPTDATA(6,""),-1) I 'ERCT G DATAX S DISPDATA(LCT)="",LCT=LCT+1 S DISPDATA(LCT)="ERROR INFORMATION:",LCT=LCT+1 </pre>	

Routine Name	GETDATA^IBCNERPE
<pre> S DISPDATA(LCT)=" F CT=1:1:ERCT D .S LCT=LCT+1,DISPDATA(LCT)="Reject Reason Code: " _\$P(RPTDATA(6,CT),U,2) ; ib*2*497 .S LCT=LCT+1,DISPDATA(LCT)="Reject Reason Text: " _\$P(RPTDATA(6,CT),U,3) ; ib*2*497 .S LCT=LCT+1,DISPDATA(LCT)="Action Code: " _\$P(RPTDATA(6,CT),U,3) .S LCT=LCT+1,DISPDATA(LCT)="HIPAA Loop: " _\$P(RPTDATA(6,CT),U,4) .S LCT=LCT+1,DISPDATA(LCT)="HL7 Location: " _\$P(RPTDATA(6,CT),U) .S LCT=LCT+1,DISPDATA(LCT)="Error Source: " _\$P(RPTDATA(6,CT),U,5) .I \$D(RPTDATA(6,CT,"AMSG")) D ..S LCT=LCT+1,DISPDATA(LCT)=" ..S LCT=LCT+1,DISPDATA(LCT)="Additional Messages:" ..S LCT=LCT+1,DISPDATA(LCT)=" ..S Z=0 F S Z=\$O(RPTDATA(6,CT,"AMSG",Z)) Q:'Z S LCT=LCT+1,DISPDATA(LCT)=RPTDATA(6,CT,"AMSG",Z) ..Q .S LCT=LCT+1,DISPDATA(LCT)=" .Q ; DATAX ; ; Disp Future Date and Misc. Comments I \$O(RPTDATA(5,0))'="" D .F CT=1:1:+\$O(RPTDATA(5,""),-1) D ..S DISPDATA(LCT)=" " _\$FO^IBCNEUT1("",7,"R")_\$G(RPTDATA(5,CT)),LCT=LCT+1 ; ; Need to add the code here that will gather the Eligibility Benefits information and place it in the ; DISPDATA array to be printed at the end of the patient's display on the e-IV Response Report. ; Q ; </pre>	

Routine Name	EBFILE^IBCNEHL1	
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change	
Requirement Traceability Matrix	2.6.5.5, 2.6.5.6	
Related Options	Process Insurance Buffer Patient Insurance Info View/Edit	
Related Routines	Routines "Called By"	Routines "Called"
	^IBCNBAR	WARN^IBCNEHL3
Data Dictionary (DD) References		

Routine Name	EBFILE^IBCNEHL1
Related Protocols	
Related Integration Control Registrations (ICRs)	
Data Passing	<input type="checkbox"/> Input <input type="checkbox"/> Output Reference <input checked="" type="checkbox"/> Both <input type="checkbox"/> Global Reference <input type="checkbox"/> Local
Input Attribute Name and Definition	
Output Attribute Name and Definition	
Current Logic	
<pre> ; EBFILE(DFN,IEN312,RIEN,AFLG) ; file eligibility/benefit data from file 365 into file 2.312 ; DFN - file 2 ien ; IEN312 - file 2.312 ien ; RIEN - file 365 ien ; AFLG - 1 if called from autoupdate, 0 if called from ins. buffer process entry ; Returns "" on success, ERFLG on failure. Also called from ACCEPT^IBCNCBAR for manual processing of ins. buffer entry. ; ; N DA,DIK,DATA,DATA1,EBIENS,ERFLG,ERROR,GIEN,GSKIP,IENROOT,IENS,IENSTR,TYPE,TYPE1, Z,Z1,Z2 ; delete existing EB data S DIK="^DPT("_DFN_",312,"_IEN312_",6,"DA(2)=DFN,DA(1)=IEN312 S DA=0 F S DA=\$O(^DPT(DFN,312,IEN312,6,DA)) Q:DA=""!(DA?1.A) D ^DIK ; file new EB data S IENSTR=IEN312_"_",_DFN_"", S GIEN=+\$P(\$G(^DPT(DFN,312,IEN312,0)),U,18) S Z="" F S Z=\$O(^IBCN(365,RIEN,2,"B",Z)) Q:Z=""!\$G(ERFLG) D .S EBIENS=\$O(^IBCN(365,RIEN,2,"B",Z,""))_","_RIEN_"", .; if filing Medicare Part A/B data, make sure we only file the correct EB group .S GSKIP=0 I GIEN>0 D ..S TYPE=\$\$GET1^DIQ(365.02,EBIENS,.05) ..S TYPE1=\$P(\$G(^IBA(355.3,GIEN,0)),U,14) ..I TYPE="MA",TYPE1="B" S GSKIP=1 ..I TYPE="MB",TYPE1="A" S GSKIP=1 ..Q .I GSKIP Q ; wrong Medicare Part A/B EB group - skip it .D GETS^DIQ(365.02,EBIENS,"*",,"DATA","ERROR") I \$D(ERROR) D:AFLG WARN^IBCNEHL3 Q .; make sure we have data to file .I '\$D(DATA(365.02)) Q .S IENS="+1,"_IENSTR,Z1=\$O(DATA(365.02,"")) M DATA1(2.322,IENS)=DATA(365.02,Z1) .D UPDATE^DIE("E","DATA1","IENROOT","ERROR") I \$D(ERROR) D:AFLG WARN^IBCNEHL3 Q .S IENS="+1,"_IENROOT(1),"_IENSTR K DATA1,IENROOT .S Z2="" F S Z2=\$O(DATA(365.26,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.3226,IENS)=DATA(365.26,Z2) D UPDATE^DIE("E","DATA1","ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.27,Z2)) Q:Z2=""!\$G(ERFLG) D </pre>	

Routine Name	EBFILE^IBCNEHL1
	<pre> ..M DATA1(2.3227,IENS)=DATA(365.27,Z2) D UPDATE^DIE("E","DATA1",,"ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.28,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.3228,IENS)=DATA(365.28,Z2) D UPDATE^DIE("E","DATA1",,"ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.29,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.3229,IENS)=DATA(365.29,Z2) D UPDATE^DIE("E","DATA1",,"ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.291,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.32291,IENS)=DATA(365.291,Z2) D UPDATE^DIE("E","DATA1",,"ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.292,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.32292,IENS)=DATA(365.292,Z2) D UPDATE^DIE("E","DATA1",,"ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .K DATA .Q Q \$G(ERFLG) ; </pre>
	<p>Modified Logic (Changes are in bold)</p> <pre> ; EBFILE(DFN,IEN312,RIEN,AFLG) ; file eligibility/benefit data from file 365 into file 2.312 ; DFN - file 2 ien ; IEN312 - file 2.312 ien ; RIEN - file 365 ien ; AFLG - 1 if called from autoupdate, 0 if called from ins. buffer process entry ; Returns "" on success, ERFLG on failure. Also called from ACCEPT^IBCNBAR for manual processing of ins. buffer entry. ; ; N DA,DIK,DATA,DATA1,EBIENS,ERFLG,ERROR,GIEN,GSKIP,IENROOT,IENS,IENSTR,TYPE,TYPE1, Z,Z1,Z2 ; delete existing EB data S DIK=^DPT("_DFN_",312,"_IEN312_",6,"DA(2)=DFN,DA(1)=IEN312 S DA=0 F S DA=\$O(^DPT(DFN,312,IEN312,6,DA)) Q:DA=""!(DA?1.A) D ^DIK ; ; Place a call here to an internal function that will file the data for the new Requested Service Date field ; (New Position TBD in node 8 of file #2.312) and the new Requested Service Type field (New Field TBD ; in node 8 of file #2.312). ; ; file new EB data S IENSTR=IEN312_"_",_DFN_"", S GIEN=+\$P(\$G(^DPT(DFN,312,IEN312,0)),U,18) S Z="" F S Z=\$O(^IBCN(365,RIEN,2,"B",Z)) Q:Z=""!\$G(ERFLG) D .S EBIENS=\$O(^IBCN(365,RIEN,2,"B",Z,""))_"_",_RIEN_"", ; if filing Medicare Part A/B data, make sure we only file the correct EB group </pre>

Routine Name	EBFILE^IBCNEHL1
<pre> .S GSKIP=0 I GIEN>0 D ..S TYPE=\$\$GET1^DIQ(365.02,EBIENS,.05) ..S TYPE1=\$P(\$G(^IBA(355.3,GIEN,0)),U,14) ..I TYPE="MA",TYPE1="B" S GSKIP=1 ..I TYPE="MB",TYPE1="A" S GSKIP=1 ..Q .I GSKIP Q ; wrong Medicare Part A/B EB group - skip it .D GETS^DIQ(365.02,EBIENS,"*",,"DATA","ERROR") I \$D(ERROR) D:AFLG WARN^IBCNEHL3 Q ; make sure we have data to file .I '\$D(DATA(365.02)) Q .S IENS="+1,"_IENSTR,Z1=\$O(DATA(365.02,"")) M DATA1(2.322,IENS)=DATA(365.02,Z1) .D UPDATE^DIE("E","DATA1","IENROOT","ERROR") I \$D(ERROR) D:AFLG WARN^IBCNEHL3 Q .S IENS="+1,"_IENROOT(1),"_IENSTR K DATA1,IENROOT .S Z2="" F S Z2=\$O(DATA(365.26,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.3226,IENS)=DATA(365.26,Z2) D UPDATE^DIE("E","DATA1","ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.27,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.3227,IENS)=DATA(365.27,Z2) D UPDATE^DIE("E","DATA1","ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.28,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.3228,IENS)=DATA(365.28,Z2) D UPDATE^DIE("E","DATA1","ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.29,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.3229,IENS)=DATA(365.29,Z2) D UPDATE^DIE("E","DATA1","ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.291,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.32291,IENS)=DATA(365.291,Z2) D UPDATE^DIE("E","DATA1","ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .S Z2="" F S Z2=\$O(DATA(365.292,Z2)) Q:Z2=""!\$G(ERFLG) D ..M DATA1(2.32292,IENS)=DATA(365.292,Z2) D UPDATE^DIE("E","DATA1","ERROR") K DATA1 I \$D(ERROR) D:AFLG WARN^IBCNEHL3 ..Q .K DATA .Q Q \$G(ERFLG) ; </pre>	

3.6.3 Data Dictionaries

File Name and Number	Eligibility/Benefit File (#2.312)
Enhancement Category	<input type="checkbox"/> New <input checked="" type="checkbox"/> Modify <input type="checkbox"/> Delete <input type="checkbox"/> No Change
Requirements Traceability Matrix	2.6.5.5, 2.6.5.6, 2.6.5.7, 2.6.5.8
Related Options	Process Insurance Buffer Third Party Joint Inquiry Patient Insurance Info View/Edit eIV Response Report
Data Dictionary (DD) References	
Related Protocols	
Related Integration Control Registrations (ICRs) Agreements	
File Documentation	
File Auditing, Security, and Archiving	

Field Name	Requested Service Date
Field Description	Requested Service Date
Field #	2.312
Node #	8
Piece #	1
New Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Data Type	<input checked="" type="checkbox"/> Date/Time <input type="checkbox"/> Numeric <input type="checkbox"/> Set of Codes <input type="checkbox"/> Free Text <input type="checkbox"/> Pointer to a File <input type="checkbox"/> Variable-Pointer
Identifier	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Uneditable Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Mandatory Field	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Field Documentation or Help Changes Necessary	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Field Definition	Requested Service Date is loaded during update process from the (#365) file in the EBFIELD^IBCNEHL1 routine.
Input/Output Transform	
Cross-Reference (id and type)	<input type="checkbox"/> Regular <input type="checkbox"/> Kwic <input type="checkbox"/> Mnemonic <input type="checkbox"/> Mumps
No cross reference	<input type="checkbox"/> Soundex <input type="checkbox"/> Trigger <input type="checkbox"/> Bulletin

Field Name	Requested Service Type
Field Description	Requested Service Type
Field #	2.312
Node #	8
Piece #	2
New Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Data Type	<input type="checkbox"/> Date/Time <input type="checkbox"/> Numeric <input type="checkbox"/> Set of Codes <input type="checkbox"/> Free Text <input checked="" type="checkbox"/> Pointer to a File <input type="checkbox"/> Variable-Pointer
Identifier	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Uneditable Field	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Mandatory Field	<input type="checkbox"/> Yes <input checked="" type="checkbox"/> No
Field Documentation or Help Changes Necessary	<input checked="" type="checkbox"/> Yes <input type="checkbox"/> No
Field Definition	Requested Service Type is loaded during update process from the (#365) file in the EBFIELD^IBCNEHL1 routine. This is a pointer to the Service Type File (#365.013).
Input/Output Transform	
Cross-Reference (id and type)	<input type="checkbox"/> Regular <input type="checkbox"/> Kwic <input type="checkbox"/> Mnemonic <input type="checkbox"/> Mumps <input type="checkbox"/> Soundex <input type="checkbox"/> Trigger <input type="checkbox"/> Bulletin
No cross reference	

4 Attachment A - Approval Signatures

This section is used to document the approval of the Software Design Document during the Formal Review. The review should be ideally conducted face to face where signatures can be obtained 'live' during the review however the following forms of approval are acceptable:

1. Physical signatures obtained face to face or via fax
2. Digital signatures tied cryptographically to the signer
3. /es/ in the signature block provided that a separate digitally signed e-mail indicating the signer's approval is provided and kept with the document

Signed:

Date:

< Integrated Project Team (IPT) Chair >



RE REQUEST FOR
VA DEV APPROVAL A(

[Redacted Signature]

8/16/13

Signed:

Date:

< Business Sponsor >

Signed:

Date:

< IT Program Manager >

Signed:

Date:

< Project Manager >

Signed:

Date:

< Technical and Enterprise Architectural Review Team >