**MCCF EDI TAS US124**

System Design Document



Department of Veterans Affairs

**December 2017**

Version 1.01

**User Story Number:** USRX-44

**User Story Name:** Transmit Coordination of Benefit Fields on Secondary Claims

**Product Backlog ID:** 44

**Priority:**

**Initial Sizing Estimate:** 13

**Rational ID:**

**Rally ID:** US124

**Epic Taxonomy:** eBusiness Compliance

# Design – Summary

When a secondary claim is sent, the system is not currently sending some COB fields which are required for TRICARE secondary claims. Logic will be added to include these fields on all secondary claims if the fields are on the payer sheet.

* 353-NR OTHER PAYER-PAT RESP AMT CNT
* 351-NP OTHER PAYER-PAT RESP AMT QLFR
* 352-NQ OTHER PAYER-PAT RESP AMOUNT

In this document, these three fields will be referred to as the Other Payer Patient Responsibility fields. All three fields appear, if at all, on segment 160, Coordination of Benefits/Other Payments.

Two of those fields (AMOUNT and AMT QLFR) will be added to the CRI (Claim Response Inquiry) screen. To facilitate testing, the incoming field PATIENT PAY AMOUNT will be added to the ECME Testing Tool. This is the value that is sent on the secondary claim as field OTHER PAYER-PAT RESP AMOUNT.

List of Components:

* File: BPS PAYER RESPONSE OVERRIDES
* File: BPS REQUESTS
* File: BPS TRANSACTION
* Routine: BPSFLD01
* Routine: BPSOSCD
* Routine: BPSOSHF
* Routine: BPSOSRB
* Routine: BPSOSRX3
* Routine: BPSPRRX3
* Routine: BPSPRRX6
* Routine: BPSTEST

# Design – Detail

This story calls for taking a field from the response to a primary claim and sending it on the secondary claim request. The system already does this with other fields. For example, the amount paid on the primary claim comes in on the response as field 509-F9, Total Amount Paid, and is sent on the secondary claim request as field 431-DV, Other Payer Amount Paid. To accomplish this, the following happens.

* The system saves the field in the BPS RESPONSES file when the response is received and processed.
* When the secondary claim is initiated, the system puts the field on the BPS REQUEST.
* From there, it is copied to the BPS TRANSACTION.
* It is pulled from the BPS TRANSACTION and put onto the claim in BPS CLAIMS when the claim is created.
* The value stored in BPS CLAIMS is put onto the claim request that is sent to the payer.

Incoming BPS BPS BPS BPS Outgoing

Claim 🡪 RESPONSES 🡪 REQUESTS 🡪 TRAN- 🡪 CLAIMS 🡪 Claim

Response ACTION Request

The new functionality called for in this story will work in the same way. Some of the existing logic will handle the new fields without any modifications.

The organization of this SDD will generally follow the above flow, from capturing Patient Pay Amount on the incoming claim response to sending the Other Payer Patient Responsibility fields on the outgoing claim request. At the end of the SDD are sections covering the testing tool, the CRI/VER screens, and the PRO Option and RED Action.

**Fields in Each File**

Most of the files in question already have the fields necessary for this story. Two new fields will need to be added.

| File | Field | New/Existing | Length | Format |
| --- | --- | --- | --- | --- |
| Incoming Claim Response | 505-F5, Patient Pay Amount | Existing | 8 | signed numeric |
| BPS RESPONSES | PATIENT PAY AMOUNT | Existing | 8 | free text |
| BPS REQUESTS | OTHER PAYER-PATIENT RESP AMT | New | 8 | free text |
| BPS TRANSACTION | OTHER PAYER-PATIENT RESP AMT | New | 8 | free text |
| BPS CLAIMS | OTHER PAYER-PATIENT RESP AMT | Existing | 12 | free text |
| BPS CLAIMS | OTHER PAYER-PT RESP AMT QUALFR | Existing | 4 | free text |
| BPS CLAIMS | OTHER PAYER-PAT RESP AMT COUNT | Existing | 4 | free text |
| Outgoing Claim Request | 352-NQ, Other Payer-Patient Responsibility Amount | Existing | 8 | signed numeric |
| Outgoing Claim Request | 351-NP, Other Payer-Patient Responsibility Amount Qualifier | Existing | 2 | alpha-numeric |
| Outgoing Claim Request | 353-NR, Other Payer-Patient Responsibility Amount Count | Existing | 2 | numeric |

A new field will be added to file# 9002313.77, BPS REQUESTS, sub-file# 9002313.7781, OTHER PAYER AMT PAID MULTIPLE, and a new field will be added to file# 9002313.59, BPS TRANSACTION, sub-file# 9002313.59141, OTHER PAYER AMT PAID MULTIPLE.

|  |  |
| --- | --- |
| Field Attributes | Values |
| FILE | 9002313.77, BPS REQUESTS |
| SUB-FILE | 9002313.7781, OTHER PAYER AMT PAID MULTIPLE |
| FIELD NAME | OTHER PAYER-PATIENT RESP AMT |
| FIELD NUMBER | .03 |
| NODE;PIECE | ^BPS(9002313.77,D0,8,D1,1,D2,0);3 |
| FIELD TYPE | Free text |
| FIELD LENGTH | 8 |
| INPUT TRANSFORM | K:$L(X)>8!($L(X)<6) X |
| HELP PROMPT | Answer must be 6-8 characters in length. |
| DESCRIPTION | Amount that is calculated by the processor and returned to the pharmacy as the TOTAL amount to be paid by the patient to the pharmacy; the patient's total cost share, including co-payments, amounts applied to deductible, over maximum amounts, penalties, etc. |

|  |  |
| --- | --- |
| Field Attributes | Values |
| FILE | 9002313.59, BPS TRANSACTION |
| SUB-FILE | 9002313.59141, OTHER PAYER AMT PAID MULTIPLE |
| FIELD NAME | OTHER PAYER-PATIENT RESP AMT |
| FIELD NUMBER | .03 |
| NODE;PIECE | ^BPST(D0,14,D1,1,D2,0);3 |
| FIELD TYPE | Free text |
| FIELD LENGTH | 8 |
| INPUT TRANSFORM | K:$L(X)>8!($L(X)<6) X |
| HELP PROMPT | Answer must be 6-8 characters in length. |
| DESCRIPTION | Amount that is calculated by the processor and returned to the pharmacy as the TOTAL amount to be paid by the patient to the pharmacy; the patient's total cost share, including co-payments, amounts applied to deductible, over maximum amounts, penalties, etc. |

The tables below describe how the files BPS REQUESTS and BPS TRANSACTION will be added to the build.

|  |  |
| --- | --- |
| File Name | BPS REQUESTS |
| Send Full or Partial DD | PARTIAL |
| Data Dictionary Number | 9002313.77 |
| Sub-File Number | 9002313.7781 |
| Field Number | .03 |
| Update the Data Dictionary | YES |
| Send Security Codes | NO |
| Screen to Determine DD Update | N/A |
| Data Comes with File | NO |
| Site’s Data | N/A |

|  |  |
| --- | --- |
| File Name | BPS TRANSACTION |
| Send Full or Partial DD | PARTIAL |
| Data Dictionary Number | 9002313.59 |
| Sub-File Number | 9002313.59141 |
| Field Number | .03 |
| Update the Data Dictionary | YES |
| Send Security Codes | NO |
| Screen to Determine DD Update | N/A |
| Data Comes with File | NO |
| Site’s Data | N/A |

**Capturing the Incoming Patient Pay Amount**

The field Patient Pay Amount, 505-F5, is already being captured in file# 9002313.03, BPS RESPONSES, field# 505, PATIENT PAY AMOUNT. No change necessary.

**BPS REQUESTS**

The claim submission process can be initiated from many places within the system, but they all eventually call EN^BPSNCPDP. When that function is called, sometimes COB data has already been pulled and set into a local array (passed into EN^BPSNCPDP as the 16th parameter, BPSPRDAT). If that array has already been created, it will automatically include the Patient Pay Amount from the primary claim response.

(The two procedures which pass the COB array into EN^BPSNCPDP are

* SUBMCLM^BPSPRRX2
  + Called from
    - SECONDRY^BPSPRRX
      * COB array built in $$PRIMDATA^BPSPRRX6 and $$SECDATA^BPSPRRX6; see SECONDRY^BPSPRRX for calls
    - PRIMARY^BPSPRRX4
      * News the array BPSDAT in PRIMARY^BPSPRRX4
      * Never sets up the array
    - SECNOPRM^BPSPRRX5
      * COB array built in $$PRIMDATA^BPSPRRX6 and $$SECDATA^BPSPRRX6; see SECNOPRM^BPSPRRX for calls
* DOSELCTD^BPSRES
  + Calls PROMPTS^BPSRES
    - COB array built in $$PRIMDATA^BPSPRRX6 and $$SECDATA^BPSPRRX6; see PROMPTS^BPSRES for calls

(So in all instances, the COB array is set up in $$PRIMDATA^BPSPRRX6 and $$SECDATA^BPSPRRX6.

($$PRIMDATA builds COBARRAY(“OTHER PAYER”,BPSPIEN,“P”) from fields on BPS RESPONSES. $$SECDATA builds BPSPRDAT(“OTHER PAYER”,BPSPIEN,“P”) from fields on BPS TRANSACTION.)

When resubmitting a secondary claim, SECDATA^BPSPRRX6 pulls COB data from the BPS TRANSACTION entry. SECDATA^BPSPRRX6 will not need to be updated; it will pull the new field, OTHER PAYER-PAT RESP AMOUNT as written.

SECDATA+48 .. S BPSPRDAT("OTHER PAYER",COBPIEN,"P",APDIEN,0)=

$G(^BPST(IEN59SEC,14,COBPIEN,1,APDIEN,0))

MKRQST^BPSOSRX3 (called only by REQST^BPSOSRX) creates an entry in the BPS REQUESTS file. That function will be modified to pull PATIENT PAY AMOUNT from the primary claim’s response when creating the new entry in BPS REQUESTS for the secondary claim.

| Subroutine Name | **MKRQST^BPSOSRX3** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Routines Which Call This Subroutine | REQST^BPSOSRX | | | |
| Current Logic | | | | |
| MKRQST(BPREQTYP,KEY1,KEY2,MOREDATA,BPIENS78,BPCOBIND,BILLNDC,BPSKIP) ;  N BPIEN77,BPCOB,BPQ,BPIEN772,BPERRMSG,BPIEN59,BPIEN78,BPZ  N RETVAL,STAT,TYPE,RESULT,SUBMITDT,BPNOW,BPACTTYP,BP77LCK  N DUR,BPIEN771,BPCNT,BPSDUPL  ...  ; store secondary billing related data entered by the user - esg 6/8/10  S BPQ=0,BPERRMSG=""  I BPCOBIND=2 D  . N AMTIEN,BPIEN1,BPIEN2,BPIEN778,BPZ,BPZ1,BPZ2,OPAMT,OPAPQ,OPAYD,OPREJ,PIEN,REJIEN  . S PIEN=0 F S PIEN=$O(MOREDATA("OTHER PAYER",PIEN)) Q:'PIEN!BPQ D  .. S OPAYD=$G(MOREDATA("OTHER PAYER",PIEN,0)) Q:OPAYD=""  .. ;  .. ; count up the number of multiples we have in each set  .. S BPZ=0 F BPZ1=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"P",BPZ)) Q:'BPZ  .. S BPZ=0 F BPZ2=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"R",BPZ)) Q:'BPZ  .. I BPZ1,BPZ2 S BPQ=1,BPERRMSG="Can't have both payments and rejects for the same OTHER PAYER" Q  .. ;  .. ; add a new entry to subfile 9002313.778  .. S BPIEN778=$$INSITEM^BPSUTIL2(9002313.778,BPIEN77,PIEN,PIEN,"",,0)  .. I BPIEN778<1 S BPERRMSG="Can't create entry in COB OTHER PAYERS multiple of the BPS REQUESTS file",BPQ=1 Q  .. S BPERRMSG="Can't populate field in COB OTHER PAYERS multiple" ; just in case BPQ is set below  .. ;  .. ; set the rest of the pieces at this level  .. I $P(OPAYD,U,2)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.02,PIEN\_","\_BPIEN77,$P(OPAYD,U,2))<1 S BPQ=1 Q  .. I $P(OPAYD,U,3)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.03,PIEN\_","\_BPIEN77,$P(OPAYD,U,3))<1 S BPQ=1 Q  .. I $P(OPAYD,U,4)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.04,PIEN\_","\_BPIEN77,$P(OPAYD,U,4))<1 S BPQ=1 Q  .. I $P(OPAYD,U,5)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.05,PIEN\_","\_BPIEN77,$P(OPAYD,U,5))<1 S BPQ=1 Q  .. I $$FILLFLDS^BPSUTIL2(9002313.778,.06,PIEN\_","\_BPIEN77,BPZ1)<1 S BPQ=1 Q  .. I $$FILLFLDS^BPSUTIL2(9002313.778,.07,PIEN\_","\_BPIEN77,BPZ2)<1 S BPQ=1 Q  .. S BPERRMSG=""  .. ;  .. ; now loop thru the other payer payment array  .. S AMTIEN=0 F S AMTIEN=$O(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN)) Q:'AMTIEN!BPQ D  ... S OPAMT=$G(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN,0))  ... S OPAPQ=$P(OPAMT,U,2) ; 342-HC other payer amt paid qualifier (ncpdp 5.1 blank is OK)  ... S OPAMT=+OPAMT ; 431-DV other payer amt paid  ... ;  ... ; add a new entry to subfile 9002313.7781  ... S BPIEN1=$$INSITEM^BPSUTIL2(9002313.7781,PIEN\_","\_BPIEN77,OPAMT,AMTIEN,"",,0)  ... I BPIEN1<1 S BPERRMSG="Can't create entry in 9002313.7781 subfile",BPQ=1 Q  ... ;  ... ; set piece 2  ... I OPAPQ'="" I $$FILLFLDS^BPSUTIL2(9002313.7781,.02,AMTIEN\_","\_PIEN\_","\_BPIEN77,OPAPQ)<1 D  .... S BPQ=1,BPERRMSG="Can't populate .02 field in 9002313.7781 subfile"  .... Q  ... Q  .. ;  .. ; now loop thru the other payer reject array  .. S REJIEN=0 F S REJIEN=$O(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN)) Q:'REJIEN!BPQ D  ... S OPREJ=$G(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN,0)) Q:OPREJ="" Q:$P(OPREJ,U,1)=""  ... ;  ... ; add a new entry to subfile 9002313.7782  ... S BPIEN2=$$INSITEM^BPSUTIL2(9002313.7782,PIEN\_","\_BPIEN77,$P(OPREJ,U,1),REJIEN,"",,0)  ... I BPIEN2<1 S BPERRMSG="Can't create entry in 9002313.7782 subfile",BPQ=1 Q  ... Q  .. Q  . Q  ... | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| MKRQST(BPREQTYP,KEY1,KEY2,MOREDATA,BPIENS78,BPCOBIND,BILLNDC,BPSKIP) ;  N BPIEN77,BPCOB,BPQ,BPIEN772,BPERRMSG,BPIEN59,BPIEN78,BPZ  N RETVAL,STAT,TYPE,RESULT,SUBMITDT,BPNOW,BPACTTYP,BP77LCK  N DUR,BPIEN771,BPCNT,BPSDUPL  ...  ; store secondary billing related data entered by the user - esg 6/8/10  S BPQ=0,BPERRMSG=""  I BPCOBIND=2 D  . N AMTIEN,BPIEN1,BPIEN2,BPIEN778,BPZ,BPZ1,BPZ2,OPAMT,OPAPQ,OPAYD,OPPRA,OPREJ,PIEN,REJIEN  . S PIEN=0 F S PIEN=$O(MOREDATA("OTHER PAYER",PIEN)) Q:'PIEN!BPQ D  .. S OPAYD=$G(MOREDATA("OTHER PAYER",PIEN,0)) Q:OPAYD=""  .. ;  .. ; count up the number of multiples we have in each set  .. S BPZ=0 F BPZ1=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"P",BPZ)) Q:'BPZ  .. S BPZ=0 F BPZ2=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"R",BPZ)) Q:'BPZ  .. I BPZ1,BPZ2 S BPQ=1,BPERRMSG="Can't have both payments and rejects for the same OTHER PAYER" Q  .. ;  .. ; add a new entry to subfile 9002313.778  .. S BPIEN778=$$INSITEM^BPSUTIL2(9002313.778,BPIEN77,PIEN,PIEN,"",,0)  .. I BPIEN778<1 S BPERRMSG="Can't create entry in COB OTHER PAYERS multiple of the BPS REQUESTS file",BPQ=1 Q  .. S BPERRMSG="Can't populate field in COB OTHER PAYERS multiple" ; just in case BPQ is set below  .. ;  .. ; set the rest of the pieces at this level  .. I $P(OPAYD,U,2)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.02,PIEN\_","\_BPIEN77,$P(OPAYD,U,2))<1 S BPQ=1 Q  .. I $P(OPAYD,U,3)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.03,PIEN\_","\_BPIEN77,$P(OPAYD,U,3))<1 S BPQ=1 Q  .. I $P(OPAYD,U,4)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.04,PIEN\_","\_BPIEN77,$P(OPAYD,U,4))<1 S BPQ=1 Q  .. I $P(OPAYD,U,5)'="" I $$FILLFLDS^BPSUTIL2(9002313.778,.05,PIEN\_","\_BPIEN77,$P(OPAYD,U,5))<1 S BPQ=1 Q  .. I $$FILLFLDS^BPSUTIL2(9002313.778,.06,PIEN\_","\_BPIEN77,BPZ1)<1 S BPQ=1 Q  .. I $$FILLFLDS^BPSUTIL2(9002313.778,.07,PIEN\_","\_BPIEN77,BPZ2)<1 S BPQ=1 Q  .. S BPERRMSG=""  .. ;  .. ; now loop thru the other payer payment array  .. S AMTIEN=0 F S AMTIEN=$O(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN)) Q:'AMTIEN!BPQ D  ... S OPAMT=$G(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN,0))  ... S OPAPQ=$P(OPAMT,U,2) ; 342-HC other payer amt paid qualifier (ncpdp 5.1 blank is OK)  ... S OPPRA=$P(OPAMT,U,3) ; 352-NQ, Other Payer-Patient Responsibility Amount  ... S OPAMT=+OPAMT ; 431-DV other payer amt paid  ... ;  ... ; add a new entry to subfile 9002313.7781  ... S BPIEN1=$$INSITEM^BPSUTIL2(9002313.7781,PIEN\_","\_BPIEN77,OPAMT,AMTIEN,"",,0)  ... I BPIEN1<1 S BPERRMSG="Can't create entry in 9002313.7781 subfile",BPQ=1 Q  ... ;  ... ; set piece 2  ... I OPAPQ'="" I $$FILLFLDS^BPSUTIL2(9002313.7781,.02,AMTIEN\_","\_PIEN\_","\_BPIEN77,OPAPQ)<1 D  .... S BPQ=1,BPERRMSG="Can't populate .02 field in 9002313.7781 subfile"  .... Q  ... ;  ... ; set piece 3  ... I OPPRA'="" I $$FILLFLDS^BPSUTIL2(9002313.7781,.03,AMTIEN\_","\_PIEN\_","\_BPIEN77,OPPRA)<1 D  .... S BPQ=1,BPERRMSG="Can't populate .03 field in 9002313.7781 subfile"  .... Q  ... ;  ... Q  .. ;  .. ; now loop thru the other payer reject array  .. S REJIEN=0 F S REJIEN=$O(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN)) Q:'REJIEN!BPQ D  ... S OPREJ=$G(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN,0)) Q:OPREJ="" Q:$P(OPREJ,U,1)=""  ... ;  ... ; add a new entry to subfile 9002313.7782  ... S BPIEN2=$$INSITEM^BPSUTIL2(9002313.7782,PIEN\_","\_BPIEN77,$P(OPREJ,U,1),REJIEN,"",,0)  ... I BPIEN2<1 S BPERRMSG="Can't create entry in 9002313.7782 subfile",BPQ=1 Q  ... Q  .. Q  . Q  ... | | | | |

**BPS TRANSACTION**

Data is pulled from the BPS REQUESTS entry and put into the MOREDATA array in READMORE^BPSOSRX4. No change is necessary to the existing code, since the new field is on the same node already referenced in this line:

READMORE+54

... S MOREDATA("OTHER PAYER",COBPIEN,"P",APDIEN,0)=

$G(^BPS(9002313.77,BPIEN77,8,COBPIEN,1,APDIEN,0))

In SECBIL59^BPSPRRX6, data in the MOREDATA array is used to populate fields on the BPS TRANSACTION. The new field will need to be added to this code.

| Subroutine Name | **SECBIL59^BPSPRRX6** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Routines Which Call This Subroutine | INIT^BPSOSIY | | | |
| Current Logic | | | | |
| SECBIL59(MOREDATA,IEN59) ;  ; Populate secondary billing fields in BPS TRANSACTION  ; MOREDATA array filed into 9002313.59  N BPTYPE,BPSTIME,BPCOB  N AMTIEN,BPIEN1,BPIEN2,BPZ5914,BPZ,BPZ1,BPZ2,OPAMT,OPAPQ,OPAYD,OPREJ,PIEN,REJIEN,BPQ  I +$G(IEN59)=0 Q  ;  I $L($G(MOREDATA("337-4C"))) I $$FILLFLDS^BPSUTIL2(9002313.59,1204,IEN59,MOREDATA("337-4C"))<1 D LOG^BPSOSL(IEN59,$T(+0)\_"-  Cannot populate (#1204) of (#9002313.59)") ; cob other payments count  I $L($G(MOREDATA("308-C8"))) I $$FILLFLDS^BPSUTIL2(9002313.59,1205,IEN59,MOREDATA("308-C8"))<1 D LOG^BPSOSL(IEN59,$T(+0)\_"-  Cannot populate (#1205) of (#9002313.59)") ; other coverage code  ;  ; store secondary billing related data entered by the user - esg 6/14/10  S BPQ=0  S PIEN=0 F S PIEN=$O(MOREDATA("OTHER PAYER",PIEN)) Q:'PIEN!BPQ D  . S OPAYD=$G(MOREDATA("OTHER PAYER",PIEN,0)) Q:OPAYD=""  . ;  . ; count up the number of multiples we have in each set  . S BPZ=0 F BPZ1=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"P",BPZ)) Q:'BPZ  . S BPZ=0 F BPZ2=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"R",BPZ)) Q:'BPZ  . I BPZ1,BPZ2 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot have both payments and rejects for the same OTHER PAYER.") Q  . ;  . ; add a new entry to subfile 9002313.5914  . S BPZ5914=$$INSITEM^BPSUTIL2(9002313.5914,IEN59,PIEN,PIEN,"",,0)  . I BPZ5914<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Can't create entry in COB OTHER PAYERS multiple of the BPS TRANSACTION fi  le") Q  . ;  . ; set the rest of the pieces at this level  . I $P(OPAYD,U,2)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.02,PIEN\_","\_IEN59,$P(OPAYD,U,2))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(  +0)\_"-Cannot populate (#.02) of (#9002313.5914)") Q  . I $P(OPAYD,U,3)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.03,PIEN\_","\_IEN59,$P(OPAYD,U,3))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(  +0)\_"-Cannot populate (#.03) of (#9002313.5914)") Q  . I $P(OPAYD,U,4)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.04,PIEN\_","\_IEN59,$P(OPAYD,U,4))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(  +0)\_"-Cannot populate (#.04) of (#9002313.5914)") Q  . I $P(OPAYD,U,5)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.05,PIEN\_","\_IEN59,$P(OPAYD,U,5))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(  +0)\_"-Cannot populate (#.05) of (#9002313.5914)") Q  . I $$FILLFLDS^BPSUTIL2(9002313.5914,.06,PIEN\_","\_IEN59,BPZ1)<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.06)  of (#9002313.5914)") Q  . I $$FILLFLDS^BPSUTIL2(9002313.5914,.07,PIEN\_","\_IEN59,BPZ2)<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.07)  of (#9002313.5914)") Q  . ;  . ; now loop thru the other payer payment array  . S AMTIEN=0 F S AMTIEN=$O(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN)) Q:'AMTIEN!BPQ D  .. S OPAMT=$G(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN,0))  .. S OPAPQ=$P(OPAMT,U,2) ; 342-HC other payer amt paid qualifier (ncpdp 5.1 blank is OK)  .. S OPAMT=+OPAMT ; 431-DV other payer amt paid  .. ;  .. ; add a new entry to subfile 9002313.59141  .. S BPIEN1=$$INSITEM^BPSUTIL2(9002313.59141,PIEN\_","\_IEN59,OPAMT,AMTIEN,"",,0)  .. I BPIEN1<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Can't create entry in 9002313.59141 subfile") Q  .. ;  .. ; set piece 2  .. I OPAPQ'="" I $$FILLFLDS^BPSUTIL2(9002313.59141,.02,AMTIEN\_","\_PIEN\_","\_IEN59,OPAPQ)<1 D  ... S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.02) of (#9002313.59141)")  ... Q  .. Q  . ;  . ; now loop thru the other payer reject array  . S REJIEN=0 F S REJIEN=$O(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN)) Q:'REJIEN!BPQ D  .. S OPREJ=$G(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN,0)) Q:OPREJ="" Q:$P(OPREJ,U,1)=""  .. ;  .. ; add a new entry to subfile 9002313.59142  .. S BPIEN2=$$INSITEM^BPSUTIL2(9002313.59142,PIEN\_","\_IEN59,$P(OPREJ,U,1),REJIEN,"",,0)  .. I BPIEN2<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Can't create entry in 9002313.59142 subfile") Q  .. Q  . Q  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| SECBIL59(MOREDATA,IEN59) ;  ; Populate secondary billing fields in BPS TRANSACTION  ; MOREDATA array filed into 9002313.59  N BPTYPE,BPSTIME,BPCOB  N AMTIEN,BPIEN1,BPIEN2,BPZ5914,BPZ,BPZ1,BPZ2,OPAMT,OPAPQ,OPAYD,OPPRA,OPREJ,PIEN,REJIEN,BPQ  I +$G(IEN59)=0 Q  ;  I $L($G(MOREDATA("337-4C"))) I $$FILLFLDS^BPSUTIL2(9002313.59,1204,IEN59,MOREDATA("337-4C"))<1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#1204) of (#9002313.59)") ; cob other payments count  I $L($G(MOREDATA("308-C8"))) I $$FILLFLDS^BPSUTIL2(9002313.59,1205,IEN59,MOREDATA("308-C8"))<1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#1205) of (#9002313.59)") ; other coverage code  ;  ; store secondary billing related data entered by the user - esg 6/14/10  S BPQ=0  S PIEN=0 F S PIEN=$O(MOREDATA("OTHER PAYER",PIEN)) Q:'PIEN!BPQ D  . S OPAYD=$G(MOREDATA("OTHER PAYER",PIEN,0)) Q:OPAYD=""  . ;  . ; count up the number of multiples we have in each set  . S BPZ=0 F BPZ1=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"P",BPZ)) Q:'BPZ  . S BPZ=0 F BPZ2=0:1 S BPZ=$O(MOREDATA("OTHER PAYER",PIEN,"R",BPZ)) Q:'BPZ  . I BPZ1,BPZ2 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot have both payments and rejects for the same OTHER PAYER.") Q  . ;  . ; add a new entry to subfile 9002313.5914  . S BPZ5914=$$INSITEM^BPSUTIL2(9002313.5914,IEN59,PIEN,PIEN,"",,0)  . I BPZ5914<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Can't create entry in COB OTHER PAYERS multiple of the BPS TRANSACTION file") Q  . ;  . ; set the rest of the pieces at this level  . I $P(OPAYD,U,2)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.02,PIEN\_","\_IEN59,$P(OPAYD,U,2))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.02) of (#9002313.5914)") Q  . I $P(OPAYD,U,3)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.03,PIEN\_","\_IEN59,$P(OPAYD,U,3))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.03) of (#9002313.5914)") Q  . I $P(OPAYD,U,4)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.04,PIEN\_","\_IEN59,$P(OPAYD,U,4))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.04) of (#9002313.5914)") Q  . I $P(OPAYD,U,5)'="" I $$FILLFLDS^BPSUTIL2(9002313.5914,.05,PIEN\_","\_IEN59,$P(OPAYD,U,5))<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.05) of (#9002313.5914)") Q  . I $$FILLFLDS^BPSUTIL2(9002313.5914,.06,PIEN\_","\_IEN59,BPZ1)<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.06) of (#9002313.5914)") Q  . I $$FILLFLDS^BPSUTIL2(9002313.5914,.07,PIEN\_","\_IEN59,BPZ2)<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.07) of (#9002313.5914)") Q  . ;  . ; now loop thru the other payer payment array  . S AMTIEN=0 F S AMTIEN=$O(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN)) Q:'AMTIEN!BPQ D  .. S OPAMT=$G(MOREDATA("OTHER PAYER",PIEN,"P",AMTIEN,0))  .. S OPAPQ=$P(OPAMT,U,2) ; 342-HC other payer amt paid qualifier (ncpdp 5.1 blank is OK)  .. S OPPRA=$P(OPAMT,U,3) ; 352-NQ, Other Payer-Patient Responsibility Amount  .. S OPAMT=+OPAMT ; 431-DV other payer amt paid  .. ;  .. ; add a new entry to subfile 9002313.59141  .. S BPIEN1=$$INSITEM^BPSUTIL2(9002313.59141,PIEN\_","\_IEN59,OPAMT,AMTIEN,"",,0)  .. I BPIEN1<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Can't create entry in 9002313.59141 subfile") Q  .. ;  .. ; set piece 2  .. I OPAPQ'="" I $$FILLFLDS^BPSUTIL2(9002313.59141,.02,AMTIEN\_","\_PIEN\_","\_IEN59,OPAPQ)<1 D  ... S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.02) of (#9002313.59141)")  ... Q  .. ;  .. ; set piece 3  .. I OPPRA'="" I $$FILLFLDS^BPSUTIL2(9002313.59141,.03,AMTIEN\_","\_PIEN\_","\_IEN59,OPPRA)<1 D  ... S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Cannot populate (#.03) of (#9002313.59141)")  ... Q  .. ;  .. Q  . ;  . ; now loop thru the other payer reject array  . S REJIEN=0 F S REJIEN=$O(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN)) Q:'REJIEN!BPQ D  .. S OPREJ=$G(MOREDATA("OTHER PAYER",PIEN,"R",REJIEN,0)) Q:OPREJ="" Q:$P(OPREJ,U,1)=""  .. ;  .. ; add a new entry to subfile 9002313.59142  .. S BPIEN2=$$INSITEM^BPSUTIL2(9002313.59142,PIEN\_","\_IEN59,$P(OPREJ,U,1),REJIEN,"",,0)  .. I BPIEN2<1 S BPQ=1 D LOG^BPSOSL(IEN59,$T(+0)\_"-Can't create entry in 9002313.59142 subfile") Q  .. Q  . Q  Q | | | | |

**BPS CLAIMS**

COB^BPSOSCD populates portions of the BPS array with data from the BPS TRANSACTION. This will be modified to include the new field. After the BPS array is built, it is used to create an entry in BPS CLAIMS.

| Subroutine Name | **COB^BPSOSCD** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Routines Which Call This Subroutine | MEDINFO^BPSOSCD | | | |
| Current Logic | | | | |
| COB(IEN59,MEDN) ; process the COB fields and build the COB array  ; Code for Payer-Patient Responsibility and Benefit Stages multiples  ; not implemented yet (except by certification)  ;  ; build array of COB secondary claim data from the BPS Transaction file - esg - 6/16/10  N COBPIEN,APDIEN,REJIEN,DATA  K BPS("RX",MEDN,"OTHER PAYER")  ;  ; Field 337-4C COB OTHER PAYMENTS COUNT (9002313.59,1204) moved into [1] below  S BPS("RX",MEDN,"OTHER PAYER",0)=$P($G(^BPST(IEN59,12)),U,4)  ;  S COBPIEN=0 F S COBPIEN=$O(^BPST(IEN59,14,COBPIEN)) Q:'COBPIEN D  . ; Note that this will set pieces 1-7. Piece 8 is reserved for  . ; Payer-Patient Responsibility Count and is set by the certification code  . S BPS("RX",MEDN,"OTHER PAYER",COBPIEN,0)=$G(^BPST(IEN59,14,COBPIEN,0))  . ;  . ; retrieve data from other payer amount paid multiple  . S APDIEN=0 F S APDIEN=$O(^BPST(IEN59,14,COBPIEN,1,APDIEN)) Q:'APDIEN D  .. S DATA=$G(^BPST(IEN59,14,COBPIEN,1,APDIEN,0))  .. S BPS("RX",MEDN,"OTHER PAYER",COBPIEN,"P",APDIEN,0)=$P(DATA,"^",1)\_"^"\_$$GET1^DIQ(9002313.2,$P(DATA,"^",2),.01)  .. Q  . ;  . ; retrieve data from other payer reject multiple  . S REJIEN=0 F S REJIEN=$O(^BPST(IEN59,14,COBPIEN,2,REJIEN)) Q:'REJIEN D  .. S BPS("RX",MEDN,"OTHER PAYER",COBPIEN,"R",REJIEN,0)=$G(^BPST(IEN59,14,COBPIEN,2,REJIEN,0))  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| COB(IEN59,MEDN) ; process the COB fields and build the COB array  ; Code for Benefit Stages multiple not implemented yet (except by  ; certification)  ;  ; build array of COB secondary claim data from the BPS Transaction file - esg - 6/16/10  N COBPIEN,APDIEN,REJIEN,DATA  K BPS("RX",MEDN,"OTHER PAYER")  ;  ; Field 337-4C COB OTHER PAYMENTS COUNT (9002313.59,1204) moved into [1] below  S BPS("RX",MEDN,"OTHER PAYER",0)=$P($G(^BPST(IEN59,12)),U,4)  ;  S COBPIEN=0 F S COBPIEN=$O(^BPST(IEN59,14,COBPIEN)) Q:'COBPIEN D  . ; Note that this will set pieces 1-7. Piece 8 is reserved for  . ; Payer-Patient Responsibility Count and is set by the certification code  . S BPS("RX",MEDN,"OTHER PAYER",COBPIEN,0)=$G(^BPST(IEN59,14,COBPIEN,0))  . ;  . ; retrieve data from other payer amount paid multiple  . S APDIEN=0 F S APDIEN=$O(^BPST(IEN59,14,COBPIEN,1,APDIEN)) Q:'APDIEN D  .. S DATA=$G(^BPST(IEN59,14,COBPIEN,1,APDIEN,0))  .. S BPS("RX",MEDN,"OTHER PAYER",COBPIEN,"P",APDIEN,0)=  $P(DATA,"^",1)\_"^"\_$$GET1^DIQ(9002313.2,$P(DATA,"^",2),.01)  .. S BPS("RX",MEDN,"OTHER PAYER",COBPIEN,"PP",APDIEN,0)=$P(DATA,"^",3)  .. I +$P(DATA,"^",3) S $P(BPS("RX",MEDN,"OTHER PAYER",COBPIEN,"PP",APDIEN,0) ,"^",2)="06"  .. Q  . ;  . ; retrieve data from other payer reject multiple  . S REJIEN=0 F S REJIEN=$O(^BPST(IEN59,14,COBPIEN,2,REJIEN)) Q:'REJIEN D  .. S BPS("RX",MEDN,"OTHER PAYER",COBPIEN,"R",REJIEN,0)=$G(^BPST(IEN59,14,COBPIEN,2,REJIEN,0))  Q | | | | |

The “set” code for each field in the file BPS NCPDP FIELD DEFS sets the value of that field into the corresponding field in the BPS CLAIMS file. For the set code for fields 351, 352, 353, see SET351^BPSFLD01, SET352^BPSFLD01, SET353^BPSFLD01. SET353 is already correct and does not need to be modified. SET351 and SET352 will be modified, as described below.

| Subroutine Name | **SET351^BPSFLD01, SET352^BPSFLD01** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Current Logic | | | | |
| SET351 ; 351-NP Other Payer-Patient Responsibility Amount Qualifier  I '$G(BPSOPIEN)!'$G(BPSOAIEN) Q  S $P(^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,BPSOAIEN,0),U,1,2)=BPSOAIEN\_U\_BPS("X")  S ^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,"B",BPSOAIEN,BPSOAIEN)=""  S ^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,0)="^9002313.401353A^"\_BPSOAIEN\_U\_BPSOAIEN  Q  ;  SET352 ; 352-NQ Other Payer-Patient Responsibility Amount Paid  I '$G(BPSOPIEN)!'$G(BPSOAIEN) Q  S $P(^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,BPSOAIEN,0),U,3)=BPS("X")  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| SET351 ; 351-NP Other Payer-Patient Responsibility Amount Qualifier  I '$G(BPSOPIEN)!'$G(BPSCOUNT) Q  S $P(^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,BPSCOUNT,0),U,1,2)=BPSCOUNT\_U\_BPS("X")  S ^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,"B",BPSCOUNT,BPSCOUNT)=""  S ^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,0)="^9002313.401353A^"\_BPSCOUNT\_U\_BPSCOUNT  Q  ;  SET352 ; 352-NQ Other Payer-Patient Responsibility Amount Paid  I '$G(BPSOPIEN)!'$G(BPSCOUNT) Q  S $P(^BPSC(BPS(9002313.02),400,BPS(9002313.0201),337,BPSOPIEN,3,BPSCOUNT,0),U,3)=BPS("X")  Q | | | | |

The many COB fields in BPS NCPDP FIELD DEFS do not each have their own “get” code. Rather, COB^BPSOSHF pulls all the fields and executes the format and set code for each. This procedure will be modified to make sure that all three Other Payer Patient Responsibility fields are populated if there is a value in Patient Responsibility from the primary claim.

| Subroutine Name | **COB^BPSOSHF** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Routines Which Call This Subroutine | XLOOP^BPSOSCF | | | |
| Current Logic | | | | |
| COB(FORMAT,NODE,MEDN) ; COB fields processing, NODE=160  ;---------------------------------------------------------------  ; The COB data is stored in the following local array:  ;  ; BPS("RX",MEDN,"OTHER PAYER",.....  ;  ; Array built in routine BPSOSCD.  ; Special note - Overrides are not allowed on this multiple.  ; "Special" code is not accounted for either.  ;---------------------------------------------------------------  ;  N FIELD,FLD,OVERRIDE,FLAG,ORD,NCPFLD,BPD,BPD1,BPD2,PCE,BPSOPIEN,BPSOAIEN,BPSORIEN  S FLAG="FS"  ;  ; Quit if there is no data in the array  Q:'$D(BPS("RX",MEDN,"OTHER PAYER"))  ;  ; next we need to figure out which fields on this format are really  ; needed, then we will loop through and populate them  ;  D GETFLDS(FORMAT,NODE,.FIELD)  ;  ; re-sort this list by the NCPDP field#  ; NCPFLD(NCPDP FIELD#) = internal field#  K NCPFLD S ORD=0 F S ORD=$O(FIELD(ORD)) Q:'ORD S FLD=$P(FIELD(ORD),U,2) I FLD'="" S NCPFLD(FLD)=+FIELD(ORD)  ;  ; see if 337-4C is needed  S FLD=337  I $D(NCPFLD(FLD)) D  . S BPS("X")=$P($G(BPS("RX",MEDN,"OTHER PAYER",0)),U,1) ; get  . I BPS("X")="" Q  . D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  . Q  ;  ; now lets get, format and set the rest of the COB fields  S BPSOPIEN=0 F S BPSOPIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN)) Q:'BPSOPIEN D  . S BPD=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,0))  . ; Note that pieces 8 (Payer-Patient Responsibility Count) and 9 (Benefit Stage Count) are only set  . ; by Certification Code  . F PCE=1:1:9 D  .. S FLD=$S(PCE=1:337,PCE=2:338,PCE=3:339,PCE=4:340,PCE=5:443,PCE=6:341,PCE=7:471, PCE=8:353,PCE=9:392,1:0) Q:'FLD  .. I '$D(NCPFLD(FLD)) Q ; field not needed  .. I $P(BPD,U,PCE)="" Q ; data is nil  .. S BPS("X")=$P(BPD,U,PCE) ; get  .. D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . ;  . ; Now look at the other payer amount paid fields  . S BPSOAIEN=0 F S BPSOAIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"P",BPSOAIEN)) Q:'BPSOAIEN D  .. S BPD1=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"P",BPSOAIEN,0))  .. F PCE=1,2 D  ... S FLD=$S(PCE=1:431,PCE=2:342,1:0) Q:'FLD  ... I '$D(NCPFLD(FLD)) Q ; field not needed  ... I $P(BPD1,U,PCE)="" Q ; data is nil  ... S BPS("X")=$P(BPD1,U,PCE) ; get  ... D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . ;  . ; Now look at the other payer reject code fields  . S BPSORIEN=0 F S BPSORIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"R",BPSORIEN)) Q:'BPSORIEN D  .. S BPD2=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"R",BPSORIEN,0))  .. S FLD=472  .. I '$D(NCPFLD(FLD)) Q ; field not needed  .. I BPD2="" Q ; data is nil  .. S BPS("X")=BPD2 ; get  .. D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . ;  . ; Now look at the other payer-patient amount paid fields  . ; Currently, this multiple is only set by certification code  . S BPSOAIEN=0 F S BPSOAIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"PP",BPSOAIEN)) Q:'BPSOAIEN D  .. S BPD1=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"PP",BPSOAIEN,0))  .. F PCE=1,2 D  ... S FLD=$S(PCE=1:352,PCE=2:351,1:0) Q:'FLD  ... I '$D(NCPFLD(FLD)) Q ; field not needed  ... I $P(BPD1,U,PCE)="" Q ; data is nil  ... S BPS("X")=$P(BPD1,U,PCE) ; get  ... D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . ;  . ; Now look at the Benefit Stages fields  . ; Currently, this multiple is only set by certification code  . S BPSOAIEN=0 F S BPSOAIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"BS",BPSOAIEN)) Q:'BPSOAIEN D  .. S BPD1=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"BS",BPSOAIEN,0))  .. F PCE=1,2 D  ... S FLD=$S(PCE=1:394,PCE=2:393,1:0) Q:'FLD  ... I '$D(NCPFLD(FLD)) Q ; field not needed  ... I $P(BPD1,U,PCE)="" Q ; data is nil  ... S BPS("X")=$P(BPD1,U,PCE) ; get  ... D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . Q  ;  COBX ;  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| COB(FORMAT,NODE,MEDN) ; COB fields processing, NODE=160  ;---------------------------------------------------------------  ; The COB data is stored in the following local array:  ;  ; BPS("RX",MEDN,"OTHER PAYER",.....  ;  ; Array built in routine BPSOSCD.  ; Special note - Overrides are not allowed on this multiple.  ; "Special" code is not accounted for either.  ;---------------------------------------------------------------  ;  N FIELD,FLD,OVERRIDE,FLAG,ORD,NCPFLD,BPD,BPD1,BPD2,PCE,BPSOPIEN,BPSOAIEN,BPSORIEN,BPSCOUNT  S FLAG="FS"  ;  ; Quit if there is no data in the array  Q:'$D(BPS("RX",MEDN,"OTHER PAYER"))  ;  ; next we need to figure out which fields on this format are really  ; needed, then we will loop through and populate them  ;  D GETFLDS(FORMAT,NODE,.FIELD)  ;  ; re-sort this list by the NCPDP field#  ; NCPFLD(NCPDP FIELD#) = internal field#  K NCPFLD S ORD=0 F S ORD=$O(FIELD(ORD)) Q:'ORD S FLD=$P(FIELD(ORD),U,2) I FLD'="" S NCPFLD(FLD)=+FIELD(ORD)  ;  ; see if 337-4C is needed  S FLD=337  I $D(NCPFLD(FLD)) D  . S BPS("X")=$P($G(BPS("RX",MEDN,"OTHER PAYER",0)),U,1) ; get  . I BPS("X")="" Q  . D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  . Q  ;  ; now lets get, format and set the rest of the COB fields  S BPSOPIEN=0 F S BPSOPIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN)) Q:'BPSOPIEN D  . S BPD=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,0))  . ; Note that piece~~s 8 (Payer-Patient Responsibility Count) and~~ 9 (Benefit Stage Count) ~~are~~is only set by Certification Code  . F PCE=1:1:7,9 D  .. S FLD=$S(PCE=1:337,PCE=2:338,PCE=3:339,PCE=4:340,PCE=5:443,PCE=6:341, PCE=7:471,~~PCE=8:353,~~PCE=9:392,1:0) Q:'FLD  .. I '$D(NCPFLD(FLD)) Q ; field not needed  .. I $P(BPD,U,PCE)="" Q ; data is nil  .. S BPS("X")=$P(BPD,U,PCE) ; get  .. D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . ;  . ; Now look at the other payer amount paid fields  . S BPSOAIEN=0 F S BPSOAIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"P",BPSOAIEN)) Q:'BPSOAIEN D  .. S BPD1=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"P",BPSOAIEN,0))  .. F PCE=1,2 D  ... S FLD=$S(PCE=1:431,PCE=2:342,1:0) Q:'FLD  ... I '$D(NCPFLD(FLD)) Q ; field not needed  ... I $P(BPD1,U,PCE)="" Q ; data is nil  ... S BPS("X")=$P(BPD1,U,PCE) ; get  ... D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . ;  . ; Now look at the other payer reject code fields  . S BPSORIEN=0 F S BPSORIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"R",BPSORIEN)) Q:'BPSORIEN D  .. S BPD2=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"R",BPSORIEN,0))  .. S FLD=472  .. I '$D(NCPFLD(FLD)) Q ; field not needed  .. I BPD2="" Q ; data is nil  .. S BPS("X")=BPD2 ; get  .. D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . ;  . ; Now look at the other payer-patient amount paid fields  . ; ~~Currently, this multiple is only set by certification code~~  . S BPSCOUNT=0 ; initialize counter  . S BPSOAIEN=0 F S BPSOAIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"PP",BPSOAIEN)) Q:'BPSOAIEN D  .. S BPD1=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"PP",BPSOAIEN,0))  .. ;  .. ; Field 352-NQ = OTHER PAYER-PAT RESP AMOUNT  .. I '$D(NCPFLD(352)) Q ; fields not needed  .. I '+$P(BPD1,U,1) Q ; data is nil or zero  .. S BPSCOUNT=BPSCOUNT+1 ; increment counter  .. S BPS("X")=$P(BPD1,U,1) ; get  .. D XFLDCODE^BPSOSCF(NODE,NCPFLD(352),FLAG) ; format/set  .. ;  .. ; If Field 352 is populated, then populate 351 and 353.  .. ;  .. ; Field 351-NP = OTHER PAYER-PAT RESP AMT QLFR  .. S BPS("X")=$P(BPD1,U,2) ; get  .. D XFLDCODE^BPSOSCF(NODE,NCPFLD(351),FLAG) ; format/set  .. ;  .. ; Field 353-NR = OTHER PAYER-PAT RESP AMT CNT  .. S BPS("X")=BPSCOUNT ; get  .. D XFLDCODE^BPSOSCF(NODE,NCPFLD(353),FLAG) ; format/set  .. Q  . ;  . ; Now look at the Benefit Stages fields  . ; Currently, this multiple is only set by certification code  . S BPSOAIEN=0 F S BPSOAIEN=$O(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"BS",BPSOAIEN)) Q:'BPSOAIEN D  .. S BPD1=$G(BPS("RX",MEDN,"OTHER PAYER",BPSOPIEN,"BS",BPSOAIEN,0))  .. F PCE=1,2 D  ... S FLD=$S(PCE=1:394,PCE=2:393,1:0) Q:'FLD  ... I '$D(NCPFLD(FLD)) Q ; field not needed  ... I $P(BPD1,U,PCE)="" Q ; data is nil  ... S BPS("X")=$P(BPD1,U,PCE) ; get  ... D XFLDCODE^BPSOSCF(NODE,NCPFLD(FLD),FLAG) ; format/set  .. Q  . Q  ;  COBX ;  Q | | | | |

**The Outgoing Claim Request**

After the entry in BPS CLAIMS is created, the system uses the data in that entry to build the claim to be sent to the payer. Before looping through the segments and fields on the payer sheet, the system copies most of the fields from the BPS CLAIMS record into the BPS array. That is done in GETBPS5^BPSECX0 and GETBPS7^BPSECX0. It is already pulling the three Other Payer Patient Responsibility fields, so no change is necessary.

After the values in BPS CLAIMS are copied into the BPS array, the system uses the data in that array to build the claim to be sent to the payer. This is largely done in XLOOP^BPSOSH2. This subroutine loops through each field on each segment on the payer sheet, and for each field it pulls the data from BPS CLAIMS and adds it to the segment. XLOOP^BPSOSH2 calls PROCCOB^BPSOSH2 to pull the COB fields, and both of those procedures are already pulling the Patient Responsibility fields.

**ECME Testing Tool**

To facilitate testing this new functionality, we will add the field PATIENT PAY AMOUNT to the ECME Testing Tool. (The Testing Tool allows the user to modify the claim response. It is only operational in non-production environments.) That field needs to be added to file# 9002313.32, BPS PAYER RESPONSE OVERRIDES.

The table below describes how the new field is going to be added to file 9002313.32, BPS PAYER RESPONSE OVERRIDES.

|  |  |
| --- | --- |
| Field Attributes | Values |
| FILE | 9002313.32, BPS PAYER RESPONSE OVERRIDES |
| FIELD NAME | PATIENT PAY AMOUNT |
| FIELD NUMBER | 2.1 |
| NODE;PIECE | 2;10 |
| FIELD TYPE | FREE TEXT |
| FIELD LENGTH | 8 |
| INPUT TRANSFORM | K:$L(X)>8!($L(X)<1) X |
| HELP PROMPT | Answer must be 1-8 characters in length |
| DESCRIPTION | This is the override value that will be used for the payer response for billing requests. The value will be used to populate the PATIENT PAY AMOUNT (#505) field of RESPONSES (#9002313.0301) subfile of the BPS RESPONSES (#9002313.03) file. This corresponds to NCPDP code 505-F5. |

The table below describes how the file BPS PAYER RESPONSE OVERRIDES will be added to the build.

|  |  |
| --- | --- |
| File Name | BPS PAYER RESPONSE OVERRIDES |
| Send Full or Partial DD | PARTIAL |
| Data Dictionary Number | 9002313.32 |
| Field Number | 2.10 |
| Update the Data Dictionary | YES |
| Send Security Codes | NO |
| Screen to Determine DD Update | N/A |
| Data Comes with File | NO |
| Site’s Data | N/A |

The procedure GETOVER^BPSTEST will be modified to prompt the user for the value of the Patient Pay Amount when the testing tool is run.

| Subroutine Name | **GETOVER^BPSTEST** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Current Logic | | | | |
| GETOVER(KEY1,KEY2,BPSORESP,BPSWHERE,BPSTYPE,BPPAYSEQ) ;  ...  .. ; Overrides to test functionality - BPS\*1\*22  .. D PROMPT(BPSTIEN,2.09,"") ; reconciliation id  ;  W ! D PROMPT(BPSTIEN,.07,0)  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| GETOVER(KEY1,KEY2,BPSORESP,BPSWHERE,BPSTYPE,BPPAYSEQ) ;  ...  .. ; Overrides to test functionality - BPS\*1\*22  .. D PROMPT(BPSTIEN,2.09,"") ; reconciliation id  .. ;  .. D PROMPT(BPSTIEN,2.1,"") ; Patient Pay Amount  ;  W ! D PROMPT(BPSTIEN,.07,0)  Q | | | | |

The procedure SETOVER^BPSTEST will be modified to override the value of the Patient Pay Amount on the incoming claim response when the testing tool is run.

| Subroutine Name | **SETOVER^BPSTEST** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Current Logic | | | | |
| SETOVER(BPSTRANS,BPSTYPE,BPSDATA) ;  ...  .. ; If payable or duplicate, get the BPSPAID amount and file it if it  .. ; exists. Also delete any reject codes  .. I BPSSRESP="P"!(BPSSRESP="D") D  ... S BPSPAID=$$GET1^DIQ(9002313.32,BPSTIEN\_",",.04,"I")  ... I BPSPAID]"" S BPSDATA(1,509)=$$DFF^BPSECFM(BPSPAID,8) ; 509 Total amount paid  ... ;  ... K BPSDATA(1,510),BPSDATA(1,511) ; kill Reject Count (510) and Reject Code (511)  ...  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| SETOVER(BPSTRANS,BPSTYPE,BPSDATA) ;  ...  .. ; If payable or duplicate, get the BPSPAID amount and file it if it  .. ; exists. Also delete any reject codes  .. I BPSSRESP="P"!(BPSSRESP="D") D  ... ;  ... S BPSX=$$GET1^DIQ(9002313.32,BPSTIEN\_",",2.1,"I") ; 505-F5 Patient Pay Amount  ... I BPSX]"" S BPSDATA(1,"505")=$$DFF^BPSECFM(BPSX,10)  ... ;  ... S BPSPAID=$$GET1^DIQ(9002313.32,BPSTIEN\_",",.04,"I")  ... I BPSPAID]"" S BPSDATA(1,509)=$$DFF^BPSECFM(BPSPAID,8) ; 509 Total amount paid  ... ;  ... K BPSDATA(1,510),BPSDATA(1,511) ; kill Reject Count (510) and Reject Code (511)  ...  Q | | | | |

**CRI/Claim Response Inquiry Screen**

The CRI/Claim Response Inquiry will automatically display the Other Payer Patient Responsibility fields from the claim request whenever they are populated and the Patient Pay Amount on the claim response whenever it is populated. No change to existing logic is necessary. This is also true of the VER, which contains the CRI.

**PRO Option and RED Action**

Both the PRO option and the RED action display to the user a list of COB fields if the claim is secondary. These fields will be added to this display:

* Other Payer Patient Responsibility Amount Qualifier
* Other Payer Patient Responsibility Amount

Both PRO and RED rely on PRIMDATA^BPSPRRX6 and GETOPAP^BPSPRRX6 to pull the COB fields from the BPS RESPONSE of the primary claim. No change to PRIMDATA^BPSPRRX6 is necessary, but GETOPAP^BPSPRRX6 will be updated to include the new field OTHER PAYER-PATIENT RESP.

If the system is not able to pull COB fields from the primary claim, then both PRO and RED rely on SECDATA^BPSPRRX6 to pull the COB fields from the BPS TRANSACTION of the secondary claim if the system is attempting to resubmit a secondary claim. No change to SECDATA^BPSPRRX6 is necessary. As it is currently written, SECDATA^BPSPRRX6 will automatically pull the new field OTHER PAYER-PATIENT RESP AMT from the BPS TRANSACTION file after that field has been added to the file.

SECDATA+48^BPSPRRX6

.. S BPSPRDAT("OTHER PAYER",COBPIEN,"P",APDIEN,0)=

$G(^BPST(IEN59SEC,14,COBPIEN,1,APDIEN,0))

| Subroutine Name | **GETOPAP^BPSPRRX6** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Related Menu Options or ListMan Actions | PRO Option  RED Action | | | |
| Routines Which Call This Subroutine | PRIMDATA^BPSPRRX6 | | | |
| Current Logic | | | | |
| GETOPAP(BPSRESP,BPSDAT) ;  ; Get the Other Payer Amount Paid values and qualifiers  ; Input:  ; BPSRESP = IEN of BPS RESPONSE file  ; BPSDAT(N)=Array of Paid Amount^Qualifier (passed by reference)  ;  I '$G(BPSRESP) Q  I '$D(^BPSR(BPSRESP,1000)) Q  N CNT,BPS509,BPS559,BPS558,BPS523,BPS563,BPS562,BPS521,BPSQUAL,BPSAMNT,BPSTAX,BPSOAP,BPSX  S CNT=0  ; Set up D.0 fields for COB segment  S BPS509=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,500)),U,9))  ; If Total Amount Paid is a negative number, set it to zero.  ; Zero Pay amount is allowed  I BPS509<0 S BPS509=0  ;  ; Cognitive Services Qualifier/Professional Service Fee Paid  S BPS562=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,560)),U,2))  I BPS562<0 S BPS562=0  I +BPS562 S CNT=CNT+1,BPSDAT(CNT)=BPS562\_U\_"06"  ;  ; Incentive Qualifier/Incentive Amt Paid  S BPS521=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,500)),U,21))  I BPS521<0 S BPS521=0  I +BPS521 S CNT=CNT+1,BPSDAT(CNT)=BPS521\_U\_"05"  ; Subtract Incentive Qualifier from Paid Amount for Drug Benefit  S BPS509=BPS509-BPS521  ;  ; Default all Tax values to zero for negative values  S BPS559=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,550)),U,9)) ; Percentage Sales Tax Paid  I BPS559<0 S BPS559=0  S BPS558=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,550)),U,8)) ; Flat Sales Tax Paid  I BPS558<0 S BPS558=0  S BPS523=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,500)),U,23)) ; Amount Attributed to Sales Tax  I BPS523<0 S BPS523=0  ;  ; Sales Tax Qualifier  S BPSTAX=BPS559+BPS558-BPS523  I BPSTAX<0 S BPSTAX=0  I +BPSTAX S CNT=CNT+1,BPSDAT(CNT)=BPSTAX\_U\_"10"  ; Subtract Sales Tax Qualifier from Paid Amount for Drug Benefit  S BPS509=BPS509-BPSTAX  ;  ; Set OTHER AMOUNT PAID multiples  S BPS563=0 F S BPS563=$O(^BPSR(BPSRESP,1000,1,563.01,BPS563)) Q:BPS563="" D  . S BPSQUAL=$P($G(^BPSR(BPSRESP,1000,1,563.01,BPS563,1)),U,1)  . ; Quit if qualifier = 99 since there is no NCPDP mapping for this qualifier  . Q:BPSQUAL']""!(BPSQUAL=99)  . S BPSAMNT=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,563.01,BPS563,1)),U,2))  . ; Default negative amounts to zero  . I BPSAMNT<0 S BPSAMNT=0  . I $D(BPSOAP(BPSQUAL)) S BPSOAP(BPSQUAL)=BPSOAP(BPSQUAL)+BPSAMNT  . I '$D(BPSOAP(BPSQUAL)) S BPSOAP(BPSQUAL)=BPSAMNT  . ; Subtract Amount if Qualifier is 01, 02, 03, 04, 09 or 11  . I "010203040911"[BPSQUAL S BPS509=BPS509-BPSAMNT  I $D(BPSOAP) D  . S BPSX="" F S BPSX=$O(BPSOAP(BPSX)) Q:BPSX="" D  . . S CNT=CNT+1,BPSDAT(CNT)=BPSOAP(BPSX)\_U\_$$GETPDIEN(BPSX)  ; Set Drug Benefit Qualifier  I BPS509<0 S BPS509=0  S CNT=CNT+1,BPSDAT(CNT)=BPS509\_U\_$$GETPDIEN("07")  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| GETOPAP(BPSRESP,BPSDAT) ;  ; Get the Other Payer Amount Paid values and qualifiers  ; Input:  ; BPSRESP = IEN of BPS RESPONSE file  ; BPSDAT(N)= Array of Other Payer fields (passed by reference)  ; [1] Paid Amount  ; [2] Qualifier  ; [3] Other Payer Patient Responsibility Amount  ;  I '$G(BPSRESP) Q  I '$D(^BPSR(BPSRESP,1000)) Q  N CNT,BPS505,BPS509,BPS559,BPS558,BPS523,BPS563,BPS562,BPS521,BPSQUAL,BPSAMNT,BPSTAX,BPSOAP,BPSX  S CNT=0  ; Set up D.0 fields for COB segment  S BPS509=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,500)),U,9))  ; If Total Amount Paid is a negative number, set it to zero.  ; Zero Pay amount is allowed  I BPS509<0 S BPS509=0  ;  ; Cognitive Services Qualifier/Professional Service Fee Paid  S BPS562=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,560)),U,2))  I BPS562<0 S BPS562=0  I +BPS562 S CNT=CNT+1,BPSDAT(CNT)=BPS562\_U\_"06"  ;  ; Incentive Qualifier/Incentive Amt Paid  S BPS521=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,500)),U,21))  I BPS521<0 S BPS521=0  I +BPS521 S CNT=CNT+1,BPSDAT(CNT)=BPS521\_U\_"05"  ; Subtract Incentive Qualifier from Paid Amount for Drug Benefit  S BPS509=BPS509-BPS521  ;  ; Default all Tax values to zero for negative values  S BPS559=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,550)),U,9)) ; Percentage Sales Tax Paid  I BPS559<0 S BPS559=0  S BPS558=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,550)),U,8)) ; Flat Sales Tax Paid  I BPS558<0 S BPS558=0  S BPS523=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,500)),U,23)) ; Amount Attributed to Sales Tax  I BPS523<0 S BPS523=0  ;  ; Sales Tax Qualifier  S BPSTAX=BPS559+BPS558-BPS523  I BPSTAX<0 S BPSTAX=0  I +BPSTAX S CNT=CNT+1,BPSDAT(CNT)=BPSTAX\_U\_"10"  ; Subtract Sales Tax Qualifier from Paid Amount for Drug Benefit  S BPS509=BPS509-BPSTAX  ;  ; Set OTHER AMOUNT PAID multiples  S BPS563=0 F S BPS563=$O(^BPSR(BPSRESP,1000,1,563.01,BPS563)) Q:BPS563="" D  . S BPSQUAL=$P($G(^BPSR(BPSRESP,1000,1,563.01,BPS563,1)),U,1)  . ; Quit if qualifier = 99 since there is no NCPDP mapping for this qualifier  . Q:BPSQUAL']""!(BPSQUAL=99)  . S BPSAMNT=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,563.01,BPS563,1)),U,2))  . ; Default negative amounts to zero  . I BPSAMNT<0 S BPSAMNT=0  . I $D(BPSOAP(BPSQUAL)) S BPSOAP(BPSQUAL)=BPSOAP(BPSQUAL)+BPSAMNT  . I '$D(BPSOAP(BPSQUAL)) S BPSOAP(BPSQUAL)=BPSAMNT  . ; Subtract Amount if Qualifier is 01, 02, 03, 04, 09 or 11  . I "010203040911"[BPSQUAL S BPS509=BPS509-BPSAMNT  I $D(BPSOAP) D  . S BPSX="" F S BPSX=$O(BPSOAP(BPSX)) Q:BPSX="" D  . . S CNT=CNT+1,BPSDAT(CNT)=BPSOAP(BPSX)\_U\_$$GETPDIEN(BPSX)  ; Set Drug Benefit Qualifier  I BPS509<0 S BPS509=0  ; Set Patient Pay Amount  S BPS505=$$DFF2EXT^BPSECFM($P($G(^BPSR(BPSRESP,1000,1,500)),U,5))  ;  S CNT=CNT+1,BPSDAT(CNT)=BPS509\_U\_$$GETPDIEN("07")\_U\_BPS505  Q | | | | |

Both the PRO option and the RED action rely on DISPSEC^BPSPRRX3 to display the COB fields. DISPSEC^BPSPRRX3 will be updated to include the new fields.

| Subroutine Name | **DISPSEC^BPSPRRX3** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Related Menu Options or ListMan Actions | PRO Option  RED Action | | | |
| Routines Which Call This Subroutine | PROMPTS^BPSPRRX3 | | | |
| Current Logic | | | | |
| DISPSEC(BPSPRARR) ;  ; Validate and Display the current secondary insurance information and prompt to edit.  ; Input:  ; BPSPARR - Array of COB data, passed by reference  ;  N BPSPIEN,BPSCOB,BPSCOV,BPX,BPSCOV,DATA  ;  ; Other Payer IEN defaults to 1 since we don't do tertiary  S BPSPIEN=1,BPSCOB="SECONDARY"  ;  ; Get Coverage Code  S BPSCOV=$G(BPSPRARR("308-C8"))  I BPSCOV="02" S BPSCOV="02 (OTHER COVERAGE EXISTS - PAYMENT COLLECTED)"  E I BPSCOV="03" S BPSCOV="03 (OTHER COVERAGE EXISTS - THIS CLAIM NOT COVERED)"  E S BPSCOV="04 (OTHER COVERAGE EXISTS - PAYMENT NOT COLLECTED)"  ;  ; Write Data  W !!,"Data for Secondary Claim"  W !,"------------------------"  W !,"Insurance: "\_$G(BPSPRARR("INS NAME"))\_" COB: "\_BPSCOB  W !,"Rate Type: "\_$$GET1^DIQ(399.3,$G(BPSPRARR("RTYPE"))\_",",.01,,,,)  W !,"Other Coverage Code: "\_BPSCOV  W !,"Other Payer Coverage Type: 01 (PRIMARY)"  W !,"Other Payer ID Qualifier: 03 (BANK INFORMATION NUMBER (BIN))"  W !,"Other Payer ID: "\_$P($G(BPSPRARR("OTHER PAYER",BPSPIEN,0)),U,4)  W !,"Other Payer Date: "\_$$FMTE^XLFDT($P($G(BPSPRARR("OTHER PAYER",BPSPIEN,0)),U,5))  ;  ; Write Paid Amounts if previous claim if they are there  I $D(BPSPRARR("OTHER PAYER",BPSPIEN,"P")) D  . S BPX=0 F S BPX=$O(BPSPRARR("OTHER PAYER",BPSPIEN,"P",BPX)) Q:BPX="" D  . . S DATA=BPSPRARR("OTHER PAYER",BPSPIEN,"P",BPX,0)  . . W !,"Other Payer Paid Qualifier: "\_$$GET1^DIQ(9002313.2,$P(DATA,U,2),.01)\_" ("\_$$GET1^DIQ(9002313.2,$P(DATA,U,2),.02)\_")"  . . W !,"Other Payer Amount Paid: $"\_$FN($P(DATA,U,1),",",2)  ;  ; Write Reject Codes if previous claims if they are there  I $D(BPSPRARR("OTHER PAYER",BPSPIEN,"R")) D  . S BPX=0 F S BPX=$O(BPSPRARR("OTHER PAYER",BPSPIEN,"R",BPX)) Q:BPX="" D  . . W !,"Other Payer Reject Code: "\_$$TRANREJ^BPSECFM($G(BPSPRARR("OTHER PAYER",BPSPIEN,"R",BPX,0)))  Q | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| DISPSEC(BPSPRARR) ;  ; Validate and Display the current secondary insurance information and prompt to edit.  ; Input:  ; BPSPRARR - Array of COB data, passed by reference  ;  N BPSPIEN,BPSCOB,BPSCOV,BPX,BPSCOV,DATA  ;  ; Other Payer IEN defaults to 1 since we don't do tertiary  S BPSPIEN=1,BPSCOB="SECONDARY"  ;  ; Get Coverage Code  S BPSCOV=$G(BPSPRARR("308-C8"))  I BPSCOV="02" S BPSCOV="02 (OTHER COVERAGE EXISTS - PAYMENT COLLECTED)"  E I BPSCOV="03" S BPSCOV="03 (OTHER COVERAGE EXISTS - THIS CLAIM NOT COVERED)"  E S BPSCOV="04 (OTHER COVERAGE EXISTS - PAYMENT NOT COLLECTED)"  ;  ; Write Data  W !!,"Data for Secondary Claim"  W !,"------------------------"  W !,"Insurance: "\_$G(BPSPRARR("INS NAME"))\_" COB: "\_BPSCOB  W !,"Rate Type: "\_$$GET1^DIQ(399.3,$G(BPSPRARR("RTYPE"))\_",",.01,,,,)  W !,"Other Coverage Code: "\_BPSCOV  W !,"Other Payer Coverage Type: 01 (PRIMARY)"  W !,"Other Payer ID Qualifier: 03 (BANK INFORMATION NUMBER (BIN))"  W !,"Other Payer ID: "\_$P($G(BPSPRARR("OTHER PAYER",BPSPIEN,0)),U,4)  W !,"Other Payer Date: "\_$$FMTE^XLFDT($P($G(BPSPRARR("OTHER PAYER",BPSPIEN,0)),U,5))  ;  ; Write Paid Amounts if previous claim if they are there  I $D(BPSPRARR("OTHER PAYER",BPSPIEN,"P")) D  . S BPX=0 F S BPX=$O(BPSPRARR("OTHER PAYER",BPSPIEN,"P",BPX)) Q:BPX="" D  . . S DATA=BPSPRARR("OTHER PAYER",BPSPIEN,"P",BPX,0)  . . W !,"Other Payer Paid Qualifier: "\_$$GET1^DIQ(9002313.2,$P(DATA,U,2),.01)\_" ("\_$$GET1^DIQ(9002313.2,$P(DATA,U,2),.02)\_")"  . . W !,"Other Payer Amount Paid: $"\_$FN($P(DATA,U,1),",",2)  . . I $P(DATA,U,3)'="" D  . . . W !,"Other Payer Patient Responsibility Amount Qualifier:"  . . . W !?40,"06 (AMT REPORTED BY PRIOR PAYER)"  . . . W !,"Other Payer Patient Responsibility Amount: $"\_$FN($P(DATA,U,3),",",2)  ;  ; Write Reject Codes if previous claims if they are there  I $D(BPSPRARR("OTHER PAYER",BPSPIEN,"R")) D  . S BPX=0 F S BPX=$O(BPSPRARR("OTHER PAYER",BPSPIEN,"R",BPX)) Q:BPX="" D  . . W !,"Other Payer Reject Code: "\_$$TRANREJ^BPSECFM($G(BPSPRARR("OTHER PAYER",BPSPIEN,"R",BPX,0)))  Q | | | | |

**Improvement to Developer Log**

During the claim submission process, entries are added to the Developer Log to create an audit trail of the process. Since the entry in the BPS REQUESTS file exists for a short time – being created at the start of claim submission and deleted before the end – the entire contents of that file entry will be added to the Developer Log.

| Subroutine Name | **BACKGR^BPSOSRB** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Current Logic | | | | |
| . . . D LOG^BPSOSL(IEN59,$T(+0)\_"-Processing the Activated request "\_BPIEN77)  . . . D LOG^BPSOSL(IEN59,$T(+0)\_"-Dequeuing. Type is "\_TYPE) | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| . . . D LOG^BPSOSL(IEN59,$T(+0)\_"-Processing the Activated request "\_BPIEN77)  . . . D LOG77(IEN59,BPIEN77) ; Log entire contents of the request.  . . . D LOG^BPSOSL(IEN59,$T(+0)\_"-Dequeuing. Type is "\_TYPE) | | | | |

| Subroutine Name | **LOG77^BPSOSRB** | | | |
| --- | --- | --- | --- | --- |
| Enhancement Category | New | Modify |  |  |
| Current Logic | | | | |
| n/a | | | | |
| Modified Logic (Changes are highlighted) | | | | |
| LOG77(IEN59,BPIEN77) ; Log entire contents of the request.  N A  M A=^BPS(9002313.77,BPIEN77)  D LOG^BPSOSL(IEN59,$T(+0)\_"-Contents of ^BPS(9002313.77,"\_BPIEN77\_"), BPS REQUEST:")  D LOGARRAY^BPSOSL(IEN59,"A")  Q | | | | |