Problem List Refactoring

Release 1.00

Ray Group International

May 29, 2012

Abstract

As part of the Open Source EHR Refactoring project, we have worked on refactoring the Problem List code. This document is to describe what we have done in term of code changes for all interested stakeholders.

Contents

1 Introduction 1

2 Code Walkthrough 3

3 Routines changed in other packages 31

4 How to use the code 33

5 Installation 35

6 Conclusions 35

1. Introduction

The goal of our code changes is to make the VistA code more modular and readable without changing functionality. The Problem List package has been chosen because it has dependencies to a moderate number of other packages. This criterion allows our initial refactored code to be manageable, but not trivial, so that lessons learned here and tools developed can be used for future refactoring of other packages.

The first phase of Problem List refactoring is concentrated on providing a complete Problem List API that is used both by CPRS RPC tags and the scroll & roll interface. In fact, RPC tags that provide information for CPRS form a suitable API, however:

1. They have minimal error checking for input parameters.

2. Although they share some code with the scroll & roll interface, the commonalities are not clearly identified in the code; typically they have non-descriptive assumed variables.
3. There are a number of places where code is copied and pasted from the scroll & roll interface code and minimal changes are done; this resulted in the duplication of business logic/database access code.
4. There are functionalities in the scroll & roll interface that is not covered by this API.
5. Globals are accessed directly.
6. This API is actually in the Order/Entry Result Reporting Package (Routines ORQQPL*).

Scroll & roll interface code also has a good structure that is mostly based on List Manager and Protocol actions, however:

1. Business logic and database access is mixed with user interface elements such as write statements, user input, and List Manager Update calls.
2. There are a number of places where there is duplicated business logic/database access code again due to copying and pasting.
3. Globals are accessed directly.

Our main goals for this phase of refactoring were:

1. To define a complete Problem List API that can be used by both scroll & roll interface and RPC tags and accessible by other packages and applications.
2. Only allow business logic/database access for scroll & roll interface to be through this API; this excludes Fileman supplied user interface that directly updates File items.
3. Remove direct global access and replace them with Fileman DBS calls.
4. Minimize duplicated code.
5. Make the code readable and document where necessary.

2. Code Walkthrough

Several APIs were developed during the refactoring effort. They are described below.

$$ACTIVE^GMPLAPI2() – Is problem active

This extrinsic function verifies if a problem is active or not.

Format

$$ACTIVE^GMPLAPI2(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to 1 if the problem is active, 0 otherwise

GMPIFN (Required) The problem IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

None

$$CODESTS^GMPLAPI2() – Check code status

This extrinsic function checks the status of the code associated with a problem.

Format

$$ CODESTS^GMPLAPI2(.RETURN,GMPIFN,ADATE)

Input Parameters

.RETURN (Required) Set to 1 if the code is active on date passed in ADATE or current date if not passed, 0 otherwise

GMPIFN (Required) Problem IFN

ADATE (Optional) The date on which to check the status of ICD9 code

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

$$DETAIL^GMPLAPI2() – Detailed problem information

This extrinsic function returns detailed information on a problem.

Format

$$DETAIL^GMPLAPI2(.RETURN,GMPIFN,GMPLMGR,GMPROV)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is RETURN(FIELD)=Data_Internal_Format^Data_External_Format:

RETURN(.01)=diagnosis

RETURN(.02)=patient_name

…

RETURN(10,0)=number of comments

RETURN(10,I)=comment^facility

GMPIFN (Required) Problem IFN

GMPLMGR (Optional) If 1 filter comments by provider

GMPROV (Optional) Provider IFN. The comments returned will be filtered by this provider.

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

$$DETAILX^GMPLAPI2() Detailed problem information, formatted

This extrinsic function returns detailed information on a problem. It is similar to DETAIL^GMPLAPI2 but the values returned are in External Format only, and the fields have human readable names

Format

$$DETAILX^GMPLAPI2(.RETURN,GMPIFN,GMPLMGR,GMPROV,GMPMULTI)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is:

RETURN("DIAGNOSIS")=ICD Code

RETURN("PATIENT")=Patient Name

RETURN("MODIFIED")=Date Last Modified

RETURN("NARRATIVE")=Provider Narrative

RETURN("ENTERED")=Date Entered ^ Entered by

RETURN("STATUS")=Status

RETURN("PRIORITY")=Priority Acute/Chronic

RETURN("ONSET")=Date of Onset

RETURN("PROVIDER")=Responsible Provider

RETURN("RECORDED")=Date Recorded ^ Recorded by

RETURN("CLINIC")=Hospital Location

RETURN("SC")=Service Connected SC/NSC/""

RETURN("EXPOSURE") = Number of exposure factors
RETURN("EXPOSURE",X)="AGENT ORANGE"
RETURN("EXPOSURE",X)="RADIATION"
RETURN("EXPOSURE",X)="ENV CONTAMINANTS"
RETURN("EXPOSURE",X)="HEAD AND/OR NECK CANCER"
RETURN("EXPOSURE",X)="MILITARY SEXUAL TRAUMA"
RETURN("EXPOSURE",X)="COMBAT VET"
RETURN("EXPOSURE",X)="SHAD"
RETURN("COMMENT") = Number of comments
RETURN("COMMENT",CNT) = Date ^ Author ^ Text of Note

GMPIFN (Required) Problem IFN

GMPLMGR (Optional) If 1 filter comments by provider

GMPROV (Optional) Provider IFN. The comments returned will be filtered by this provider.

GMPMULTI (Optional) Multidivisional. If 0 the comments returned will be filtered by facility.

Output
A Boolean value signaling if the call was successful or not

Error Codes Returned
None

DELETE^GMPLAPI2() – Deletes a problem
This extrinsic function deletes an existing problem from the list.

Format
$$ DELETE^GMPLAPI2(.RETURN,GMPIFN,GMPROV,REASON)

Input Parameters
.RETURN (Required) Set to 0 if the delete failed, 1 otherwise. If it failed, RETURN will be will hold an array of error descriptions.

GMPIFN (Required) Problem IFN

GMPROV (Required) Provider IFN

REASON (Optional) Comment describing the reason for deleting this problem.

Output
A Boolean value signaling if the call was successful or not

Error Codes Returned
INVPARAM Invalid parameter passed (invalid GMPIFN, GMPROV or GMPVAMC)

PRBDLTD Problem already deleted

DELETED^GMPLAPI2() – Is problem deleted

This extrinsic function verifies if a problem is deleted or not.

Format

$$DELETED^GMPLAPI2(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to 1 if the problem is deleted, 0 otherwise

GMPIFN (Required) The problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

DUPL^GMPLAPI2() – Check for duplicate entries

This extrinsic function finds duplicate problem entries.

Format

$$DUPL^GMPLAPI2(.RETURN,GMPDFN,TERM,TEXT,GMPBOTH)

Input Parameters

.RETURN (Required) If duplicate problem is found this will be set to its IFN, 0 otherwise

GMPDFN (Required) Patient IFN

TERM (Required) Problem id. Pointer to the EXPRESSIONS file # 757.01

TEXT (Optional) Problem text to look for

GMPBOTH (Optional) Both TERM and TEXT should match in order to flag a duplicate.

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

INACTV^GMPLAPI2() - Inactivate a problem

This extrinsic function inactivates an existing problem.

Format

$$INACTV^GMPLAPI2(.RETURN,GMPIFN,GMPROV,NOTE,RESOLVED)

Input Parameters

.RETURN (Required) Set to 0 if the inactivation failed, 1 otherwise. If it failed, RETURN will hold an array of error descriptions.

GMPIFN (Required) Problem IFN

GMPROV (Required) Provider IFN

NOTE (Optional) Comment describing the reason for inactivating this problem.

RESOLVED (Optional) Resolved date

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

PRBDLDT Problem is deleted

PRBINACT Problem is already inactive.

NEW^GMPLAPI2() – Add new problem to list

This extrinsic function adds a new problem to the list of patient problems.

Format

$$NEW^GMPLAPI2(.RETURN,GMPDFN,GMPROV,,.GMPFLD)

Input Parameters

.RETURN (Required) Set to the new problem IFN, 0 otherwise. If it failed, RETURN will hold an array of error descriptions.

GMPDFN (Required) Patient IFN

GMPROV (Required) Provider IFN

.GMPFLD Array passed by reference that holds the new data. It should be in the following format: GMPFLD(FIELD)=Data_Internal_Format^Data_External_Format. E.g.:

GMPFLD (.01)=diagnosis

GMPFLD (.02)=patient_name

…

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

INVPARAM Invalid parameter passed (GMPDFN, GMPVAMC)

ONSET^GMPLAPI2() – Returns onset date

This extrinsic function returns the onset date of a problem.

Format

$$ONSET^GMPLAPI2(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to the onset date

GMPIFN (Required) Problem IFN

Output
A Boolean value signaling if the call was successful or not

Error Codes Returned
None

UPDATE^GMPLAPI2() – Update an existing problem
This extrinsic function updates a problem's data.

Format
$$UPDATE^GMPLAPI2(.RETURN,GMPIFN,.GMPORIG,.GMPFLD,GMPLUSER,,GMPROV)

Input Parameters
.RETURN (Required) Set to 0 if the save failed, 1 otherwise. If it failed, RETURN will hold an array of error descriptions.

GMPIFN (Required) Problem IFN

.GMPORIG (Required) Array of original values. See DETAIL^GMPLAPI2 for format

.GMPFLD (Required) Array of modified values. See DETAIL^GMPLAPI2 for format.

GMPLUSER (Optional) 0 if the call comes from an unattended session, 1 otherwise. Default 1.

GMPROV (Required) Provider IFN

Output
A Boolean value signaling if the call was successful or not

Error Codes Returned
None

VERIFIED^GMPLAPI2() – Is problem verified
This extrinsic function checks if a problem is verified or not.

Format
$$VERIFIED^GMPLAPI2(.RETURN,GMPIFN)

Input Parameters
.RETURN (Required) Set to 1 if the problem is verified, 0 otherwise

GMPIFN (Required) The problem IFN

Output
A Boolean value signaling if the call was successful or not

Error Codes Returned
None

VERIFY^GMPLAPI2() – Verify a transcribed problem

This extrinsic function marks a transcribed problem as permanent

Format

$$VERIFY^GMPLAPI2(.RETURN,GMPIFN)

Input Parameters

.RETURN	(Required) Set to 0 if the save failed, 1 otherwise. If it failed, RETURN will hold an array of error descriptions.
GMPIFN	(Required) Problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

PRBVRFD	Problem already verified
FILELOCKED	File is in use. Try again later.

NEWNOTE^GMPLAPI3() – Add new comment to problem

This extrinsic function adds a new comment to a problem.

Format

$$NEWNOTE^GMPLAPI3(.RETURN,GMPIFN,GMPROV,.NOTES)

Input Parameters

.RETURN	(Required) Set to 0 if the call failed, 1 otherwise. If it failed, RETURN will hold an array of error descriptions.
GMPIFN	(Required) Problem IFN
GMPROV	(Required) Provider IFN
.NOTES	(Required) Array passed by reference that holds comment lines. It should be in the following format: NOTES(I)=comment

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

INVPARAM	Invalid parameter passed (GMPIFN, GMPVAMC,NOTES)
PRBDLTD	Problem is deleted
ICDINACT	Inactive ICD found. Edit diagnosis first.
FILELOCK	File is in use. Try again later.

NOTES^GMPLAPI3() – Return list of comments for problem

This extrinsic function returns the list of comments assigned to a problem.

Format

$$NOTES^GMPLAPI3(.RETURN,GMPIFN,GMPACT)

Input Parameters

.RETURN	(Required) Array passed by reference that will receive the data. The output format is RETURN(I)=comment
GMPIFN	(Required) Problem IFN
GMPACT	(Optional) Active. If 1 only active comments will be returned.

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

UPDNOTE^GMPLAPI3() – Replaces an existing comment

This extrinsic function updates a problem's data.

Format

$$UPDNOTE^GMPLAPI3(.RETURN,GMPIFN,OLDNOTE,NEWNOTE, GMPROV)

Input Parameters

.RETURN	(Required) Set to 0 if the save failed, 1 otherwise. If it failed, RETURN will hold an array of error descriptions.
GMPIFN	(Required) Problem IFN
OLDNOTE	(Required) Text of the comment to be replaced
NEWNOTE	(Required) New comment formatted as: Note_IFN^Facility_IFN^Text If Text is empty the comment will be deleted.
GMPROV	(Required) Provider IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

BUILDLST^GMPLAPI4() – Return list of problems

This extrinsic function returns a list of detailed information on problem IFNs passed in GMPLIST.

Format

$$BUILDLST^GMPLAPI4(.RETURN,.GMPLIST)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN=Total number of problems in patient's file

RETURN(0)=number of problems returned

RETURN(I)=IFN^status^problem^ICD9^onset^last_modified^sc^exposures^condition^

location^loc_type^provider^service^priority^has_comments^

date_recorded^sc_condition^inactive

.GMPLIST (Required) List of problem IFNs in the following format:

GMPLIST(0)=number of records

GMPLIST(1)=IFN 1

…

GMPLIST(n)=IFN n

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

DIAG^GMPLAPI4() – Return ICD code

This extrinsic function returns ICD code for a problem.

Format

$$DIAG^GMPLAPI4(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) ICD code in the following format: pointer_to_icd_file^icd_code

GMPIFN (Required) Problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

FLDNAME^GMPLAPI4() – Return problem field names

This extrinsic function returns an array of problem field names.

Format

$$FLDNAME^GMPLAPI4(.RETURN)

Input Parameters

.RETURN (Required) Array passed by reference that will receive data. Format:

RETURN(field_no)=field_name

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

GETPLIST^GMPLAPI4() – Return list of problem IFNs

This extrinsic function returns a list of problems IFNs given the patient IFN.

Format

$$GETPLIST^GMPLAPI4(.RETURN,GMPDFN,GMPSTAT,GMPREV,GMPROV,GMPVIEW, GMPIDX)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is
RETURN=Total number of problems in patient's file

RETURN(0)=number of problems returned

RETURN(I)=IFN

GMPDFN (Required) Patient IFN

GMPSTAT (Optional) Status of problems to be returned. Can be any combination of (A)ctive, (I)nactive and (R)emoved. Default "A" – returns active problems only

GMPREV (Optional) Reversed order. The problems will be sorted in reversed order of recorded date.

GMPROV (Optional) Provider IFN. If present, the problems returned will be filtered by this provider. Default – return all problems

GMPVIEW (Optional) Filter by service location or clinic. Format "S/facility_ifn/facility_ifn/…" or "C/clinic_ifn/clinic_ifn/…". If facility IFN's are note passed, returns inpatient problems when GMPVIEW="S" or outpatient ones when it is set to "C". Default – returns all problems.

GMPIDX (Optional) Create "B" index. If set to 1 will append and index to the output array.

RETURN("B",IFN)=I. Default - 0

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

HASPRBS^GMPLAPI4() – Are any problems assigned to the patient

This extrinsic function returns a Boolean value signaling if there are any problems in the patient file.

Format

$$HASPRBS^GMPLAPI4(.RETURN,GMPDFN,GMPSTAT)

Input Parameters

.RETURN (Required) Set to 1 if patient file contains problems with status GMPSTAT

GMPDFN (Required) Patient IFN

GMPSTAT (Optional) Problem status (Active/Inactive). Default: both active and inactive.

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

LASTMOD^GMPLAPI4() – Last modified date

This extrinsic function returns the last modified date for a problem.

Format

$$LASTMOD^GMPLAPI4(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to last modified date

GMPIFN (Required) Problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

LIST^GMPLAPI4() – Return list of problems

This extrinsic function returns detailed information on a problem.

Format

$$LIST^GMPLAPI4(.RETURN,GMPDFN,GMPSTAT, GMPROV,GMPVIEW, GMPREV,GMPIDX)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN=Total number of problems in patient's file

RETURN(0)=number of problems returned

RETURN(I)=IFN^status^problem^ICD9^onset^last_modified^sc^exposures^condition^

location^loc_type^provider^service^priority^has_comments^

date_recorded^sc_condition^inactive

GMPDFN (Required) Patient IFN

GMPSTAT (Optional) Status of problems to be returned. Can be any combination of (A)ctive, (I)nactive and (R)emoved. Default "A" – returns active problems only

GMPROV (Optional) Provider IFN. If present, the problems returned will be filtered by this provider. Default – return all problems

GMPVIEW (Optional) Filter by service location or clinic. Format "S/facility_ifn/facility_ifn/…" or "C/clinic_ifn/clinic_ifn/…". If facility IFN's are note passed, returns inpatient problems when GMPVIEW="S" or outpatient ones when it is set to "C". Default – returns all problems.

GMPREV (Optional) Reversed order. The problems will be sorted in reversed order of recorded date.

GMPIDX (Optional) Create "B" index. If set to 1 will append and index to the output array.

RETURN("B",IFN)=I. Default - 0

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

PATIENT^GMPLAPI4() – Get patient IFN

This extrinsic function returns the patient IFN given problem IFN.

Format

$$PATIENT^GMPLAPI4(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to patient IFN

GMPIFN (Required) Problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

PROBNARR^GMPLAPI4() – Get provider narrative

This extrinsic function returns the provider narrative for a given problem IFN.

Format

$$PROBNARR^GMPLAPI4(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to patient narrative_ifn^problem_narrative

GMPIFN (Required) Problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

REPLACE^GMPLAPI4() – Replace diagnosis

This extrinsic function replaces the diagnosis code for a given problem IFN.

Format

$$REPLACE^GMPLAPI4(.RETURN,GMPIFN,NEWDIAG)

Input Parameters

.RETURN (Required) Set 1 if the call succeeded

GMPIFN (Required) Problem IFN

NEWDIAG (Required) The new ICD code

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

UNDELETE^GMPLAPI4() – Undeletes problem

This extrinsic function marks a deleted problem as permanent.

Format

$$UNDELETE^GMPLAPI4(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to 1 if the call succeeded

GMPIFN (Required) Problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

INVREC Invalid problem IFN

VALID^GMPLAPI4() – Is problem IFN valid

This extrinsic function returns a boolean value signaling if the problem IFN corresponds to a valid record.

Format

$$VALID^GMPLAPI4(.RETURN,GMPIFN)

Input Parameters

.RETURN (Required) Set to 1 if problem file contains GMPIFN

GMPIFN (Required) Problem IFN

Output

A Boolean value signaling if the call was successful or not

Error Codes Returned

None

$$NEWLST^GMPLAPI1() – Add new list

Add new problem selection list.

Format

NEWLST^GMPLAPI1(.RETURN,GMPLST,GMPLLOC)

Input Parameters

.RETURN (Required) Set to the new problem selection list IFN if succeed, 0 otherwise

GMPLLST (Required) The problem selection list name

GMPLLOC (Optional) IFN of location which will be assigned to the new problem selection list

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (list name must be 3-30 characters, not numeric or starting with punctuation, GMPLLOC is not a valid IEN)

LOCNFND Location does not exist

LISTXST Another list with the same name already exists

$$ASSUSR^GMPLAPI6() – Assign list to users

Assign a problem selection list to one or more users.

Format

ASSUSR^GMPLAPI6(.RETURN,GMPLST,GMPLUSER)

Input Parameters

.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLLST (Required) The problem selection list IFN

GMPLUSER (Required) Users IFN list separated by "^"

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLLST or GMPLUSER)

LISTNFND Problem selection list not found

PROVNFND Provider not found

$$REMUSR^GMPLAPI6() – Remove list from users

Remove assigned problem selection list from one or more users.

Format

REMUSR^GMPLAPI6(.RETURN,GMPLST,GMPLUSER)

Input Parameters

.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLLST (Required) The problem selection list IFN

GMPLUSER (Required) Users IFN list separated by "^"

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLLST or GMPLUSER)

LISTNFND Problem selection list not found

PROVNFND Provider not found

$$DELLST^GMPLAPI1() – Delete list

Delete a problem selection list if this list is not assigned to any users.

Format

DELLST^GMPLAPI1(.RETURN,GMPLST)

Input Parameters

.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLLST (Required) The problem selection list IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLLST)

LISTNFND Problem selection list not found

LISTUSED List is assigned to at least one user

$$LSTUSED^GMPLAPI1() – Is list used

Verifies if a problem selection list is assigned to some users.

Format

LSTUSED^GMPLAPI1(.RETURN,GMPLST)

Input Parameters

.RETURN (Required) Set to 1 if list is assigned to at least one user, 0 otherwise

GMPLLST (Required) The problem selection list IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLLST)

LISTNFND Problem selection list not found

$$NEWCAT^GMPLAPI1() – Add new category

Add new problem category.

Format

NEWCAT^GMPLAPI1(.RETURN,GMPGRP)

Input Parameters

.RETURN (Required) Set to the new problem category IFN if succeed, 0 otherwise

GMPLGRP (Required) The problem category name

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (category name must be 3-30 characters, not numeric or starting with punctuation)

CTGEXIST Another problem category with the same name already exists

$$DELCAT^GMPLAPI1() – Delete category

Delete a problem category if it is not used by any problem selection list.

Format

DELCAT^GMPLAPI1(.RETURN,GMPGRP)

Input Parameters

.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLGRP (Required) The problem category IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (category name must be 3-30 characters, not numeric or starting with punctuation)

CTGNFND Problem category not found

CATUSED If there is at least one problem selection list that contains the category problem

$$CATUSED^GMPLAPI1() – Is category used

Verifies if a problem category is used by any problem selection list.

Format

CATUSED^GMPLAPI1(.RETURN,GMPGRP)

Input Parameters

.RETURN (Required) Set to 1 if problem category is used by at least one list, 0 otherwise

GMPLGRP (Required) The problem category IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (category name must be 3-30 characters, not numeric or starting with punctuation)

CTGNFND Problem category not found

$$LOCKLST^GMPLAPI1() – Lock list

Lock specified problem selection list.

Format

LOCKLST^GMPLAPI1(.RETURN,GMPLST)

Input Parameters

.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLLST (Required) The problem selection list IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLLST)

LISTNFND Problem selection list not found

FILELOCK Problem selection list is already locked by another process

$$UNLKLST^GMPLAPI1() – Unlock list

Unlock specified problem selection list.

Format

UNLKLST^GMPLAPI1(GMPLST)

Input Parameters

GMPLLST (Required) The problem selection list IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

None

$$LOCKCAT^GMPLAPI1() – Lock category

Lock specified problem category.

Format

LOCKCAT^GMPLAPI1(.RETURN,GMPGRP)

Input Parameters

.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLGRP (Required) The problem category IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLGRP)

CTGNFND Problem category not found

FILELOCK Problem category is already locked by another process

$$UNLKCAT^GMPLAPI1() – Unlock category

Unlock specified problem category.

Format

UNLKCAT^GMPLAPI1(GMPLGRP)

Input Parameters

GMPLGRP (Required) The problem selection list IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

None

$$GETLIST^GMPLAPI1() – Detailed problem list information

Returns specified problem selection list details.

Format

$$GETLIST^GMPLAPI1(.RETURN,GMPLLST,CODLEN)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN("LST","NAME") – problem selection list name

RETURN("LST","MODIFIED") - date last modified

RETURN(0) - number of categories

RETURN(List Content IEN)= Sequence ^ Group IEN ^ Group name ^ Show problems flag

RETURN("GRP", Category IEN)=List Content IEN

RETURN("SEQ",# Sequence)=List Content IEN

RETURN("GRP", Category IEN,# Sequence)=Problem name ^ Problem code ^ Inactive flag (1 for inactive code, 0 for active)

GMPLLST (Required) The problem selection list IFN

CODLEN (Optional) A number that specifies the maxim length of the returned problem text

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLLST)

LISTNFND Problem selection list not found

$$GETCAT^GMPLAPI1() – Detailed category information

Returns specified problem selection category.

Format

$$GETCAT^GMPLAPI1(.RETURN,GMPLGRP)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN(Problem IEN)=Sequence ^ Pointer to problem(757.01) ^ Display text ^ ICD Code

RETURN(Problem IEN, "CODE")=ICD Code ^ Inactive flag

RETURN("SEQ", Sequence #)=Problem IEN

RETURN("PROB", Pointer to Problem(757.01))=Problem IEN

GMPLGRP (Required) The problem category IFN

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLGRP)

CTGNFND Problem category not found

$$GETCATD^GMPLAPI5() – Detailed problems information

Returns specified category problems details.

Format

$$GETCATD^GMPLAPI5(.RETURN,GMPLGRP)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN("GRP", Category IEN,# Sequence)=Problem name ^ Problem code ^ Inactive flag

GMPLGRP (Required) The problem category IFN

CODLEN (Optional) A number that specifies the maxim length of the returned problem text

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLGRP)

CTGNFND Problem category not found

$$SAVLST^GMPLAPI1() – Save list

Save changes to existing list.

Format

$$SAVLST^GMPLAPI1(.RETURN,GMPLLST,SOURCE)

Input Parameters

.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLLST (Required) The problem selection list IFN

SOURCE Contains collection of problem categories that will be assigned to the selection list

SOURCE(n)=Sequence ^ Category IEN ^ Category name ^ Show problems flag

If n is of the form “0001N” that category will be added to the list contents, else the existing category will be modified. SOURCE(n)=”@” specifies that this category will be removed from selection list contents.

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned
INVPARAM Invalid parameter passed (invalid GMPLLST)

LISTNFND Problem selection list not found

$$SAVGRP^GMPLAPI1() – Save category
Save changes to existing problem category.

Format
$$SAVGRP^GMPLAPI1(.RETURN,GMPLGRP,SOURCE)

Input Parameters
.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLGRP (Required) The problem selection list IFN

SOURCE Contains collection of problems that will be assigned to the problem category

SOURCE(n)=Sequence ^ Category IEN ^ Problem IEN ^ Problem text ^ ICD code

If n is of the form "0001N" that problem will be added to the category contents, else the existing problem will be modified. SOURCE(n)="@" specifies that this problem will be removed from category contents.

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned
INVPARAM Invalid parameter passed (invalid GMPLGRP)

CTGNFND Problem category not found

$$ADDLOC^GMPLAPI5() – Add location
Assigns a clinic to a problem selection list.

Format
$$ADDLOC^GMPLAPI5(.RETURN,GMPLLST,GMPLLOC)

Input Parameters
.RETURN (Required) Set to 1 if operation succeed, 0 otherwise

GMPLLST (Required) The problem selection list IFN

GMPLLOC (Required) The clinic location IFN

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLLST or GMPLLOC)

LISTNFND Problem selection list not found

LOCNFND Location not found

$$GETLSTS^GMPLAPI5() – Get lists

This function returns all existing problems selection lists, or those that match the search criteria.

Format

$$GETLSTS^GMPLAPI5(.RETURN,SEARCH,START,NUMBER)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN(0)=number of entries found ^ maximum requested ^ any more?

The number of entries found will be equal to or less than the maximum requested

The maximum requested should be equal the NUMBER parameter or if NUMBER not passed, "*".

The any more? Is 1 if there are more matching entries, or 0 if not.

RETURN(I,"ID")=problem selection list IFN

RETURN(I,"NAME")=problem selection list name

SEARCH (Optional) The partial match restriction.

.START (Optional) The index from which to begin the list. It can be used for pagination, passed by reference will be set to the last entry returned, if NUMEBER parameter is specified and there are more matching entries. Subsequent will use this parameter to know where to start the next list.

NUMBER (Optional) The number of entries to return.

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

None

$$GETCATS^GMPLAPI5() – Get categories

This function returns all existing problems categories, or those that match the search criteria.

Format

$$GETCATS^GMPLAPI5(.RETURN,SEARCH,START,NUMBER)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN(0)=number of entries found ^ maximum requested ^ any more?

The number of entries found will be equal to or less than the maximum requested

The maximum requested should be equal the NUMBER parameter or if NUMBER not passed, "*".

The any more? Is 1 if there are more matching entries, or 0 if not.

RETURN(I,"ID")=problem category IFN

RETURN(I,"NAME")=problem category name

SEARCH (Optional) The partial match restriction.

.START (Optional) The index from which to begin the list. It can be used for pagination, passed by reference will be set to the last entry returned, if NUMEBER parameter is specified and there are more matching entries. Subsequent will use this parameter to know where to start the next list.

NUMBER (Optional) The number of entries to return.

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

None

$$GETUSRS^GMPLAPI5() – Get users

This function returns all existing users, or those that match the search criteria.

Format

$$GETUSRS^GMPLAPI5(.RETURN,SEARCH,START,NUMBER)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN(0)=number of entries found ^ maximum requested ^ any more?

The number of entries found will be equal to or less than the maximum requested

The maximum requested should be equal the NUMBER parameter or if NUMBER not passed, "*".

The any more? Is 1 if there are more matching entries, or 0 if not.

RETURN(I,"ID")=user IFN

RETURN(I,"NAME")=user's name

RETURN(I,"INITIAL")=user's initial

RETURN(I,"EMAIL")=user's email

RETURN(I,"WRITE")=type of user

SEARCH (Optional) The partial match restriction.

.START (Optional) The index from which to begin the list. It can be used for pagination, passed by reference will be set to the last entry returned, if NUMEBER parameter is specified and there are more matching entries. Subsequent will use this parameter to know where to start the next list.

NUMBER (Optional) The number of entries to return.

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

None

$$GETCLIN^GMPLAPI5() – Get clinics

This function returns all existing clinics, or those that match the search criteria.

Format

$$GETCLIN^GMPLAPI5(.RETURN,SEARCH,START,NUMBER)

Input Parameters

.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN(0)=number of entries found ^ maximum requested ^ any more?

The number of entries found will be equal to or less than the maximum requested

The maximum requested should be equal the NUMBER parameter or if NUMBER not passed, "*".

The any more? Is 1 if there are more matching entries, or 0 if not.

RETURN(I,"ID")=location IFN

RETURN(I,"NAME")=location name

SEARCH (Optional) The partial match restriction.

.START (Optional) The index from which to begin the list. It can be used for pagination, passed by reference will be set to the last entry returned, if NUMEBER parameter is specified and there are more matching entries. Subsequent will use this parameter to know where to start the next list.

NUMBER (Optional) The number of entries to return.

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned
None

$$GETASUSR^GMPLAPI5() – Get problem selection list assigned users
This function returns the users assigned to the problem selection list.

Format
$$GETASUSR^GMPLAPI5(.RETURN,GMPLST)

Input Parameters
.RETURN (Required) Array passed by reference that will receive the data. The output format is

RETURN(0)=number of currently assigned users

RETURN(I,"ID")=problem category IFN

RETURN(I,"NAME")=problem category name

GMPLST (Required) The problem selection list IFN.

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned
INVPARAM Invalid parameter passed (invalid GMPLST)

LISTNFND Problem selection list not found

$$GETULST^GMPLAPI6() – Get problem selection list assigned to the user
This function returns the problem selection list assigned to the user.

Format
$$GETULST^GMPLAPI6(.RETURN,USER)

Input Parameters
.RETURN (Required) Passed by reference, will receive the data. The output format is

Problem selection list IFN ^ problem selection list name

USER (Required) User IFN.

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned
INVPARAM Invalid parameter passed (invalid USER)

PROVNFND Provider not found

$$GETCLST^GMPLAPI6() – Get first problem selection list assigned to the clinic

This function returns the first problem selection list assigned to the clinic.

Format

$$GETCLST^GMPLAPI6(.RETURN,GMPCLIN)

Input Parameters

.RETURN (Required) Passed by reference, will receive the data. The output format is

Problem selection list IFN ^ problem selection list name

GMPCLIN (Required) Location IFN.

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPCLIN)

LOCNFND Location not found

$$VALGRP^GMPLAPI6() – Check category for inactive codes

This function checks all problems in the category for inactive codes.

Format

$$VALGRP^GMPLAPI6(.RETURN,GMPLCAT)

Input Parameters

.RETURN (Required) Passed by reference, will be set to 1 if category has no problems with inactive codes, 0 if has one or more.

GMPLCAT (Required) Category IFN.

Output

A Boolean value signaling if the call was successful or not.

Error Codes Returned

INVPARAM Invalid parameter passed (invalid GMPLCAT)

CTGNFND Category not found

$$VALLIST^GMPLAPI6() – Check selection list for inactive codes

This function checks all categories in the list for problems with inactive codes.

Format

$$VALLIST^GMPLAPI6(.RETURN,LIST)

Input Parameters

.RETURN (Required) Passed by reference, will be set to 1 if list has no problems with inactive codes, 0 if has one or more.

LIST (Required) Problem selection list IFN.

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned
INVPARAM Invalid parameter passed (invalid LIST)

LISTNFND Problem selection list not found

$$GET^GMPLSITE() – Get site parameters
Gets the site Problem List parameters.

Format
$$GET^GMPLSITE(.RETURN)

Input Parameters
.RETURN (Required) Set to site parameters.

RETURN("VER") – Verify problems

RETURN("PRT") – Prompt for chart

RETURN("CLU") – Use Clinical Lexicon Incorrect

RETURN("REV") – Display Order incorrect

RETURN("SDP") – Screen duplicate entries.

Output
None.

Error Codes Returned
None

$$SET^GMPLSITE() – Set site parameters
Sets the site Problem List parameters.

Format
$$SET^GMPLSITE(.RETURN,.PARAM)

Input Parameters
RETURN (Required) Set to error messages if there are errors.

PARAM (Required) Input values. Same scubsripts as in GET^GMPLSITE

Output
A Boolean value signaling if the call was successful or not.

Error Codes Returned
INVPARAM Invalid parameter passed (invalid PARAM)

PPROBCNT^GMPLAPI7(RETURN) – Patients with problems

Gets the patient with problems.

Format

$$PPROBCNT^GMPLAPI7(.RETURN)

Input Parameters

RETURN (Required) Patients with problems

RESULT Number of patients.

RESULT("<Patient name>)"=<number of active problems>_"^"_<number of inactive problems>

Output

None.

Error Codes Returned

None

3. Routines changed in other packages

During the refactoring effort, some routines belonging to other packages than Problem List were modified. A summary of those routines and of what has changed is outlined below.

In most cases, direct access to globals owned by Problem List were replaced with cals to their API equivalents. The RPC Broker entries in Order Entry/Results Reporting packaged served as a basis for the refactoring process, so they were changed to go through the new API's.

We also added a new Mumps code item HIDE (54) to PROTOCOL file. Using this item you can specify condition to hide a Protocol item. GMPL Verify (1532) now has this item specified so that it is hidden when "Verify Transcribed Problems" is false and vice versa. Protocol update during changing "Verify Transcribed Problems" is thus removed.

Tests have been developed to exercise these changes. Changes in Order Entry/Results Reporting package are covered by Sikuli CPRS tests as well as M-Unit tests (^ZZRGUTRB). For the other packages, M-Unit tests are available (^ZZRGUTEX).

Automated Info Collection Sys

IBDFBK3 - AICS broker Utilities

PROBNAR^IBDFBK3	Modified to call PROBNARR^GMPLAPI4 instead of directly retrieving data from ^AUTNPOV
PROBDIA^IBDFBK3	Modified to call DIAG^GMPLAPI4 instead of directly retrieving data from ^AUPNPROB

IBDFN11 - ENCOUNTER FORM - (entry points for reprint of dynamic data)

REPRINT^IBDFN11	Retrieves diagnosis data through DIAG^GMPLAPI4 instead of ^AUPNPROB

Clinical Case Registries

RORHL17 - HL7 PROBLEM LIST: OBR,OBX

EN1^RORHL17	Retrieves the list of patient problems by calling GETPLIST^GMPLAPI4
LOAD^RORHL17	Gets problem details by calling DETAIL^GMPLAPI2

Clinical Reminders

PXRMISE - Index size estimating routines.

NEPROB^PXRMISE	Returns number of entries in PROBLEM LIST through a call to PRBCNT^GMPLAPI4

PXRMPROB - Code for Problem List.

OUTPUT^PXRMPROB	Retrieves provider narrative through a call to PROBNARR^GMPLAPI4

Kernel

XQOR3 - Process Menus, Protocol Menus

MENU^XQOR3	Added initialization for a new XQORM subscript (“R”).

XQORM1 - Display selections & prompt

DISP^XQORM1	Added functionality to hide the menu if XQORM(“R”) is not defined

Order Entry/Results Reporting

ORCPROB - Problem List interface

VERIFY^ORCPROB	Replaced direct access to ^AUPNPROB with checking the GMPSAVED return value from VERIFY^GMPL1

ORQQPL1 - PROBLEM LIST FOR CPRS GUI

EDLOAD^ORQQPL1	Replaced call to GETFLDS^GMPLEDT3 with call to the equivalent routine DETAIL^GMPLAPI2
EDSAVE^ORQQPL1	Replaced direct access to ^AUPNPROB and call to GMPLSAVE with the equivalent refactored API entry UPDATE^GMPLAPI2
ADDSAVE^ORQQPL1	Replaced direct access to ^AUPNPROB and call to GMPLSAVE with the equivalent refactored API entry NEW^GMPLAPI2
INITUSER^ORQQPL1	Duplicate code to retrieve site parameters moved to the refactored API entry GET^GMPLSITE
DUP^ORQQPL1	Replaced direct access to ^GMPL(125.99) with calls to the refactored API entries DUPL^GMPLAPI2 and ACTIVE^GMPLAPI2

ORQQPL2 - RPCs FOR CPRS GUI IMPLEMENTATION

HIST^ORQQPL2	Replaced direct access to ^GMPL(125.8) file with equivalent calls to GETHIST^GMPLHIST and AUDET^GMPLHIST
DELETE^ORQQPL2	Moved the whole logic and direct access to ^AUPNPROB to the API entry DELETE^GMPLAPI2
REPLACE^ORQQPL2	Same as above for UNDELETE^GMPLAPI4
VERIFY^ORQQPL2	Moved business logic to the API entry VERIFY^GMPLAPI2
INACT^ORQQPL2	Moved business logic to INACTV^GMPLAPI2
GETCOMM^ORQQPL2	Business logic moved and consolidated to NOTES^GMPLAPI3

ORQQPL3 - Problem List RPCs

LIST^ORQQPL3	Duplicate code to retrieve site parameters moved to the

	refactored API entry GET^GMPLSITE, main business logic moved to LIST^GMPLAPI4
DELLIST^ORQQPL3	Business logic moved and consolidated to LIST^GMPLAPI4
CAT^ORQQPL3	Data retrieval now done in a call to GETLIST^GMPLAPI1 instead of direct global access
GETUSLST^ORQQPL3	Business logic moved to GETULST^GMPLAPI6 and GETCLST^GMPLAPI6
PROB^ORQQPL3	Data retrieval logic moved to GETCAT^GMPLAPI1

PCE Patient Care Encounter

PXCAPL - Validates data from the PCE Device Interface into a call to update Problem List

PROBLEM^PXCAPL	Calls VALID^GMPLAPI4 to find if Problem is in file, calls PATIENT^GMPLAPI4 to find if a certain problem is associated to a certain patient.

PXCAPOV - Validates data from the PCE Device Interface into PCE's PXK format for POV

DIAG^PXCAPOV	Calls VALID^GMPLAPI4 to find if Problem is in file, calls PATIENT^GMPLAPI4 to find if a certain problem is associated to a certain patient.

QUASAR

ACKQUTL6 - Utilities routine

PLIST^ACKQUTL6	Retrieves list of patient problems and their status by calling GETPLIST^GMPLAPI4 and DETAIL^GMPLAPI2 respectively

4. How to use the code

API

The APIs presented in this paper cover most of the functionality found in the Problem List package. They follow a consistent calling convention that allows for returning rich error messages from the called routines, making them suitable as an interface to be used by external applications.

Assumed Variables

The only assumed variables used in this API are Kernel variables documented in section 2.3.1.3.2 of the SAC. In this version the following assumed variables were used: DUZ,DT,U

Format and conventions of the calls

The conventions used in this API are very similar to those enforced by the RPC Broker. Every tag has at least one parameter, passed by reference that will hold the result of the call. Every function returns a Boolean to signal if the processing was successful or not.

If the return value is 0, the first parameter (RETURN) will be structured as an array containing the errors encountered, numbered from 0 to the error count. The errors have the following form:

ErrorId^Message

Example

INVPARAM^Invalid parameter value – GMPIFN

5. Installation

Along with this paper a KID Host File is provided. The steps required to install the distribution are outlined below:

1. From the *Systems Manager Menu* select *Programmer Options…*
2. Select *Kernel Installation & Distribution System*
3. Select *Installation*
4. Select *Load a Distribution*
5. Enter the host file path, for example C:\ GMPL_2.0_260002.KID and load the distribution.
6. Use the *Install Package(s)* option and select GMPL*2.0*260002
7. When prompted *Want KIDS to Rebuild Menu Trees Upon Completion of Install?* Respond NO
8. When prompted *Want KIDS to INHIBIT LOGONs during the install?* Respond NO
9. When prompted *Want to DISABLE Scheduled Options, Menu Options, and Protocols?* Respond NO

6. Conclusions

This paper presented a set of APIs developed as part of the Open Source HER refactoring effort. They are meant to be used by both scroll & roll interface and RPC tags and to be accessible by other packages and applications. Also they should remove direct global access, uncouple the business logic from the user interface elements and minimize code duplication.