Contents

[ Sample **ITK** program : delete all **bomview** revisions 3](#_Toc49165221)

[ Sample Teamcenter C++ function : prevent deletion of **BOMView** Revision with children 5](#_Toc49165222)

[ Sample ITK function : create bop window 6](#_Toc49165223)

[ Sample ITK program : report configuration rule 7](#_Toc49165224)

[code to delete the BOM line from structure based on attribute value. 8](#_Toc49165225)

[ Sample **ITK** function : demo manufacturing BOM link design to ebom 9](#_Toc49165226)

[ Sample **ITK** function : demo undo child line assign 10](#_Toc49165227)

[ Sample **ITK** function : action handler to add all structure children to target list 11](#_Toc49165228)

[ Sample **ITK** program : create occurrence effectivity 13](#_Toc49165229)

[ No **ITK** function found to remove Child Module Constraints 17](#_Toc49165230)

[ Sample **ITK** function : write plmxml file 18](#_Toc49165231)

[ Sample **ITK** program : create product structure 19](#_Toc49165232)

[ Sample **ITK** program : set item revision master form attribute based on bom attributes 23](#_Toc49165233)

[ Sample **ITK** program : remove all bomline substitutes 26](#_Toc49165234)

[ Sample **ITK** program : print custom bom view 28](#_Toc49165235)

[ Sample **ITK** function : ask item revision from bom line 31](#_Toc49165236)

[ Sample **ITK** program : remove all bomline substitutes 32](#_Toc49165237)

[ Revision Rule - **ITK** 34](#_Toc49165238)

[Creation and management of product configurator objects through **ITK** customization 35](#_Toc49165239)

[Sample Teamcenter C++ function : new root item from template using saved **variant** **rule** 37](#_Toc49165240)

[Sample Teamcenter C++ function : new child item from template using saved **variant** **rule** 39](#_Toc49165241)

[Sample **ITK** function : create saved **variant** **rule** 41](#_Toc49165242)

[Process template filter / Conditions when launching New Process with several Types 42](#_Toc49165243)

[DETAILS 42](#_Toc49165244)

[SOLUTION 42](#_Toc49165245)

[ Sample ITK function : create saved variant rule 47](#_Toc49165246)

[Sample ITK function : create modular variant persistent sos 48](#_Toc49165247)

[Creation and management of product configurator objects through **ITK** customization 49](#_Toc49165248)

[ Sample Teamcenter C++ function : new root item from template using saved **variant** **rule** 51](#_Toc49165249)

[ Sample Teamcenter C++ function : new child item from template using saved **variant** **rule** 53](#_Toc49165250)

[Sample **ITK** function : list **secondary** objects 54](#_Toc49165251)

[Sample **ITK** function : va list for GRM copy 55](#_Toc49165252)

[Sample **ITK** function : demo removal of certain reference types 56](#_Toc49165253)

[Sample **ITK** function : relation create pre action 58](#_Toc49165254)

[Creation and management of product configurator objects through customization 59](#_Toc49165255)

[ Sample **ITK** function : report referencers of workspaceobject 61](#_Toc49165256)

[ Sample **ITK** function : find item revisions completed jobs 62](#_Toc49165257)

[Sample ITK function : create relation with required property 63](#_Toc49165258)

#  Sample **ITK** program : delete all **bomview** revisions

/\*HEAD DELETE\_ALL\_BOMVIEW\_REVISIONS CCC **ITK** \*/

#include <stdlib.h>

#include <tccore/aom.h>

#include <tc/emh.h>

#include <tc/tc.h>

#include <sa/tcfile.h>

#include <tccore/item.h>

#include <**itk**/mem.h>

#include <pom/pom/pom.h>

#define EXIT\_FAILURE 1

#define ERROR\_CHECK(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, (X)))

static void report\_error( char \*file, int line, char \*function, int return\_code)

{

if (return\_code != ITK\_ok)

{

char \*error\_message\_string;

EMH\_get\_error\_string (NULLTAG, return\_code, &error\_message\_string);

printf ("ERROR: %d ERROR MSG: %s.\n", return\_code, error\_message\_string);

printf ("FUNCTION: %s\nFILE: %s LINE: %d\n", function, file, line);

if(error\_message\_string) MEM\_free(error\_message\_string);

printf("\nExiting program!\n");

exit (EXIT\_FAILURE);

}

}

#define EXIT\_IF\_NULL(X) (check\_value(#X, (X)))

static void check\_value( char \*function, int value )

{

if ((value == 0) || (value == 0))

{

printf ("\t%s is NULL\n", function);

printf("\nExiting program!\n");

exit (EXIT\_FAILURE);

}

}

static void do\_it(void)

{

logical

\*is\_it\_null = NULL,

\*is\_it\_empty = NULL;

int

rc = ITK\_ok,

n\_bvrs = 0,

ii = 0,

n\_values = 0;

tag\_t

rev = NULLTAG,

\*bvrs = NULL,

attr\_id = NULLTAG,

\*attr\_tags = NULL;

char

\*name = NULL;

ERROR\_CHECK( ITEM\_find\_rev("5397460", "P", &rev) );

EXIT\_IF\_NULL( rev );

ERROR\_CHECK( AOM\_lock(rev) );

ERROR\_CHECK( POM\_attr\_id\_of\_attr( "structure\_revisions",

"ItemRevision", &attr\_id) );

EXIT\_IF\_NULL( attr\_id );

ERROR\_CHECK( POM\_length\_of\_attr( rev, attr\_id, &n\_values) );

EXIT\_IF\_NULL(n\_values);

ERROR\_CHECK(POM\_ask\_attr\_tags(rev, attr\_id, 0, n\_values, &attr\_tags,

&is\_it\_null, &is\_it\_empty ));

EXIT\_IF\_NULL(n\_values);

for (ii = 0; ii < n\_values; ii++)

ERROR\_CHECK( POM\_remove\_from\_attr( 1, &rev, attr\_id, ii, 1 ) );

if(n\_values < 0) MEM\_free(attr\_tags);

if(n\_values < 0) MEM\_free(is\_it\_null);

if(n\_values < 0) MEM\_free(is\_it\_empty);

ERROR\_CHECK( AOM\_save( rev ));

ERROR\_CHECK( AOM\_refresh( rev, FALSE ));

ERROR\_CHECK( ITEM\_rev\_list\_bom\_view\_revs(rev, &n\_bvrs, &bvrs) );

EXIT\_IF\_NULL( n\_bvrs );

for (ii = 0; ii < n\_bvrs; ii++)

{

ERROR\_CHECK( AOM\_ask\_name(bvrs[ii], &name) );

ERROR\_CHECK( AOM\_delete( bvrs[ii]) );

printf("- Deleting %s\n", name);

MEM\_free(name);

}

if (n\_bvrs < 0) MEM\_free(bvrs);

}

int ITK\_user\_main(int argc, char\* argv[])

{

int

status = 0;

ITK\_initialize\_text\_services( ITK\_BATCH\_TEXT\_MODE );

status = ITK\_auto\_login();

if ( (status != ITK\_ok)) printf("\nLogin Failed!\n\n");

else

{

printf("\nLogin successful!\n\n");

ITK\_set\_journalling(TRUE);

do\_it();

}

ITK\_exit\_module(TRUE);

return status;

}

#  Sample Teamcenter C++ function : prevent deletion of **BOMView** Revision with children

/\*

\* Business Modeler Extension Rule Definition:

\* Business Object Name: ItemRevision

\* Business Object or Property: Property

\* Property Name: structure\_revisions

\* Operation Name: setStructure\_revisions

\* Extension Point: PreCondition

\*/

#include <A2gtac/A2\_prevent\_deletion\_of\_**BOMView**\_Revision\_with\_children.hxx>

#include <A4gtac/A4customerrors\_error.h>

#include <iostream>

#include <bom/bom\_errors.h>

#include <tc/tc.h>

#include <tc/tc\_startup.h>

#include <tccore/aom\_prop.h>

#include <base\_utils/IFail.hxx>

#include <base\_utils/ScopedSmPtr.hxx>

#include <base\_utils/TcResultStatus.hxx>

using namespace Teamcenter;

using namespace std;

int A2\_prevent\_deletion\_of\_**BOMView**\_Revision\_with\_children( METHOD\_message\_t \* msg, va\_list /\*args\*/)

{

printf("\n %s \n", \_\_FUNCTION\_\_);

int ifail = **ITK**\_ok;

ResultStatus rstat;

try

{

tag\_t rev\_tag = msg->object\_tag;

int n\_children = 0;

scoped\_smptr<tag\_t> children;

rstat = AOM\_ask\_value\_tags(rev\_tag, "view", &n\_children, &children);

if (n\_children > 0)

{

EMH\_store\_error\_s1(EMH\_severity\_error, 950001,

"Cannot delete BVR with children");

ifail = 950001;

}

}

catch( const IFail &e )

{

printf("\n\n\n");

cout << "error " << e.ifail() << endl;

cout << e.getMessage() << endl;

}

return ifail;

}

#  Sample ITK function : create bop window

#include <**bom**/bom.h>

#include <cfm/cfm.h>

#include <me/me.h>

#include <ps/ps.h>

#include <tccore/aom.h>

#include <tccore/item.h>

static void create\_bop\_window(tag\_t tProcess, tag\_t tProcessRevision, tag\_t tOperation, tag\_t tOperationRevision)

{

IFERR\_REPORT(AOM\_refresh(tProcess, TRUE));

tag\_t tWindow = NULLTAG;

IFERR\_REPORT(ME\_create\_bop\_window(&tWindow));

tag\_t tRule = NULLTAG;

IFERR\_REPORT(CFM\_find("Latest Working", &tRule));

IFERR\_REPORT(BOM\_set\_window\_config\_rule(tWindow, tRule));

tag\_t tTopLine = NULLTAG;

IFERR\_REPORT(BOM\_set\_window\_top\_line(tWindow, tProcess, NULLTAG, NULLTAG, &tTopLine));

int iNumBVs = 0;

tag\_t \*ptBVs = NULL;

IFERR\_REPORT(ITEM\_list\_bom\_views(tProcess, &iNumBVs, &ptBVs));

if (iNumBVs == 0)

{

tag\_t tBv = NULLTAG;

IFERR\_REPORT(PS\_create\_bom\_view (NULLTAG, "", "", tProcess, &tBv));

IFERR\_REPORT(AOM\_save(tBv));

tag\_t tBvr = NULLTAG;

IFERR\_REPORT(PS\_create\_bvr (tBv, "", "", false, tProcessRevision, &tBvr));

IFERR\_REPORT(AOM\_save (tBvr));

IFERR\_REPORT(AOM\_save (tProcess));

}

if(ptBVs) MEM\_free(ptBVs);

tag\_t tChildLine = NULLTAG;

IFERR\_REPORT(BOM\_line\_add(tTopLine, tOperation, tOperationRevision, NULLTAG, &tChildLine));

IFERR\_REPORT(BOM\_save\_window(tWindow));

IFERR\_REPORT(BOM\_close\_window(tWindow));

IFERR\_REPORT(AOM\_refresh(tProcess, FALSE));

IFERR\_REPORT(AOM\_unload(tProcess));

}

#  Sample ITK program : report configuration rule

#include <stdlib.h>

#include <tc/tc.h>

#include <sa/tcfile.h>

#include <tccore/workspaceobject.h>

#include <ae/ae.h>

#include <user\_exits/user\_exits.h>

#include <ss/ss\_const.h>

#include <tccore/item.h>

#include <pom/pom/pom.h>

#include <ae/dataset.h>

#include <tccore/tctype.h>

#include <tccore/tc\_msg.h>

#define ITK\_CALL(x) { \

int stat; \

char \*err\_string; \

if( (stat = (x)) != ITK\_ok) \

{ \

EMH\_get\_error\_string (NULLTAG, stat, &err\_string); \

printf ("ERROR: %d ERROR MSG: %s.\n", stat, err\_string); \

printf ("FUNCTION: %s\nFILE: %s LINE: %d\n",#x, \_\_FILE\_\_, \_\_LINE\_\_); \

if(err\_string) MEM\_free(err\_string); \

exit (EXIT\_FAILURE); \

} \

}

#define EXIT\_FAILURE 1

static void do\_it(void)

{

char

\*rule\_name;

tag\_t

rule,

window;

ITK\_CALL(BOM\_create\_window (&window));

ITK\_CALL(BOM\_ask\_window\_config\_rule(window, &rule));

ITK\_CALL(CFM\_**ask\_rule\_text**(rule, &rule\_name));

printf("---%s\n", rule\_name);

if (rule\_name) MEM\_free(rule\_name);

}

int ITK\_user\_main(int argc, char\* argv[])

{

int

status = 0;

char

\*message;

ITK\_initialize\_text\_services( 0 );

status = ITK\_auto\_login();

if ( (status != ITK\_ok)) printf("iMAN login NOT successful.\n");

else

{

printf("iMAN login successful.\n");

ITK\_set\_journalling(TRUE);

do\_it();

}

ITK\_exit\_module(TRUE);

return status;

}

# code to delete the BOM line from structure based on attribute value.

 Symptom

 To remove the child **BOM** **line** comparing "**bom** **line** item id" attribute from BOM using **ITK** programming.

 Hardware/Software Configuration

 Platform: INTEL

OS: WINDOWS

OS Version: 10\_1507

Product: TEAMCENTER

Application: CUSTOMIZATION

Version: V11.4.0.2

Function: **ITK**

 Solution

 TK\_initialize\_text\_services(0);  
  
  
 ifail = ITK\_init\_module( pszUserName , pszPassword , pszGroup );  
  
 ifail = ITEM\_find\_item("000754", &itemTag);   
 ifail = ITEM\_find\_revision(itemTag, "A", &itemRevTag);

 ifail = ITEM\_rev\_list\_bom\_view\_revs(itemRevTag, &n\_bvrs, &bvrs);   
 printf("n\_bvrs is %d \n",n\_bvrs);  
 ifail = BOM\_create\_window(&window);  
 ifail = BOM\_set\_window\_top\_line(window, NULLTAG, itemRevTag, NULLTAG, &top\_line) ;  
 ifail = BOM\_line\_ask\_child\_lines(top\_line, &n\_children, &children);  
 for (ii = 0; ii < n\_children; ii++)  
 {  
 ifail = AOM\_ask\_value\_string(children[ii], "bl\_item\_item\_id", &value);  
 printf("%s \n", value);  
 if (!strcmp(value, "000756"))   
 {  
 ifail = BOM\_line\_cut(children[ii]);}  
 }  
 ifail = BOM\_save\_window(window);

 Reference

 Here in above code we are comparing the id, and cut it.e.g. 000756 id **bom** **line** will get cut after running this code.

#  Sample **ITK** function : demo manufacturing BOM link design to ebom

/\*

Define and Assign extension rule on

BOMLine - BOM\_link\_design\_to\_ebom - BaseAction

\*/

#include <ug\_va\_copy.h>

#include <me/me.h>

int S4\_BOM\_link\_design\_to\_ebom( METHOD\_message\_t\* msg, va\_list args )

{

int ifail = ITK\_ok;

va\_list largs;

va\_copy( largs, args );

tag\_t tDesignLine = va\_arg(args, tag\_t); //Design **BOM** **Line**

tag\_t tPartLine = va\_arg(args, tag\_t); //Part **BOM** **Line**

int iAlignMode = va\_arg(args, int); // Alignment Mode

va\_end( largs );

/\*

Insert your custom logic before making alignment

and finally do alignment

\*/

ifail = ME\_align\_design\_bom(tDesignLine, tPartLine, iAlignMode);

return ifail;

}

#  Sample **ITK** function : demo undo child line assign

#include <A2gtac/A2assignChildLinePostAction.hxx>

#include <string.h>

#include <ug\_va\_copy.h>

#include <tccore/aom\_prop.h>

#include <bom/bom.h>

/\* disable unreferenced formal parameter errors \*/

#pragma warning(disable: 4100)

int A2assignChildLinePostAction( METHOD\_message\_t \*msg, va\_list args )

{

printf("\n BOMLine - fnd0assignChildLine - PostAction \n");

int ifail = **ITK**\_ok;

/\* va\_list for fnd0assignChildLine \*/

va\_list largs;

va\_copy (largs, args) ;

tag\_t srcLine = va\_arg(largs, tag\_t);

const char \*occTypeName = va\_arg(largs, const char\*);

tag\_t \*pastedLine = va\_arg(largs, tag\_t \*);

va\_end(largs);

tag\_t targetLine = msg->object\_tag;

char\* targetLineName = NULL;

ifail = AOM\_ask\_value\_string(targetLine, "object\_string", &targetLineName);

printf(" Target parent line: %s\n", targetLineName);

MEM\_free(targetLineName);

char\* srcLineName = NULL;

ifail = AOM\_ask\_value\_string(srcLine, "object\_string", &srcLineName);

printf(" Source line to be pasted: %s\n", srcLineName);

MEM\_free(srcLineName);

printf(" Occtype = %s\n", occTypeName?occTypeName:"NULL");

char\* pastedLineName = NULL;

ifail = AOM\_ask\_value\_string(\*pastedLine, "object\_string", &pastedLineName);

printf(" Pasted line: %s\n", pastedLineName);

MEM\_free(srcLineName);

/\* Just to show that you can undo the previous. \*/

ifail = **BOM\_line**\_cut(\*pastedLine);

if ( ifail == **ITK**\_ok )

{

tag\_t winTag = NULLTAG;

ifail = **BOM\_line**\_ask\_window(msg->object\_tag, &winTag);

ifail = BOM\_save\_window(winTag);

}

return ifail;

}

#  Sample **ITK** function : action handler to add all structure children to target list

#include <bom/bom.h>

#include <epm/epm.h>

#include <epm\epm\_toolkit\_tc\_utils.h>

#include <tccore/aom.h>

#include <tccore/aom\_prop.h>

#include <user\_exits/epm\_toolkit\_utils.h>

static void get\_all\_target\_revs(tag\_t tTopLine, counted\_tag\_list\_t \*ptlNewTargets)

{

int ifail = **ITK**\_ok;

int iNumChildren = 0;

tag\_t \*ptChildren = NULL;

ifail = **BOM\_line**\_ask\_child\_lines(tTopLine, &iNumChildren, &ptChildren);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

for(int ii = 0; ii < iNumChildren; ii++)

{

tag\_t tRev = NULLTAG;

ifail = AOM\_ask\_value\_tag(ptChildren[ii], "bl\_revision", &tRev);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

ifail = EPM\_\_add\_to\_tag\_list(tRev, ptlNewTargets);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

/\* call recursively \*/

get\_all\_target\_revs(ptChildren[ii], ptlNewTargets);

}

MEM\_free(ptChildren);

}

extern int add\_all\_components\_to\_target\_list(EPM\_action\_message\_t msg)

{

int ifail = **ITK**\_ok;

tag\_t tRootTask = NULLTAG;

ifail = EPM\_ask\_root\_task(msg.task, &tRootTask);

int iNumAttachs = 0;

tag\_t \*ptAttachs = NULL;

ifail = EPM\_ask\_attachments(tRootTask, EPM\_target\_attachment, &iNumAttachs, &ptAttachs);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

tag\_t tTopRev = ptAttachs[0]; /\* assuming just one \*/

tag\_t window = NULLTAG;

ifail = BOM\_create\_window (&window);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

tag\_t tTopLine = NULLTAG;

ifail = BOM\_set\_window\_top\_line(window, NULLTAG, tTopRev, NULLTAG, &tTopLine);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

counted\_tag\_list\_t tlNewTargets = {0};

int initial\_tag\_list\_size = 16; // Reference PR-8968371

tlNewTargets.list = (tag\_t \*)MEM\_alloc(initial\_tag\_list\_size \* (sizeof(tag\_t)));

get\_all\_target\_revs(tTopLine, &tlNewTargets);

ifail = BOM\_close\_window(window);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

int \*piAttachTypes = NULL;

piAttachTypes = (int \*) MEM\_alloc (tlNewTargets.count \* sizeof(int));

for (int ii = 0; ii < tlNewTargets.count; ii++)

{

piAttachTypes[ii] = EPM\_target\_attachment;

}

ifail = AOM\_refresh(tRootTask, TRUE);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

ifail = EPM\_add\_attachments(tRootTask, tlNewTargets.count, tlNewTargets.list, piAttachTypes);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

if(piAttachTypes) MEM\_free(piAttachTypes);

ifail = AOM\_save(tRootTask);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

return ifail;

}

#  Sample **ITK** program : create occurrence effectivity

/\*HEAD CREATE\_OCCURRENCE\_EFFECTIVITY CCC **ITK** \*/

#ifdef \_\_cplusplus

extern "C" {

#endif

#include <stdlib.h>

#include <stdarg.h>

#include <string.h>

#include <tccore/aom.h>

#include <tccore/aom\_prop.h>

#include <tccore/item.h>

#include <bom/bom.h>

#include <**itk**/mem.h>

#include <tc/tc.h>

#include <tccore/item.h>

#include <bom/bom.h>

#include <cfm/cfm.h>

#include <ps/ps\_errors.h>

#include <tc/emh.h>

#include <tc/tc.h>

#include <ps/ps.h>

#define ITEM\_find\_rev GTAC\_find\_rev

#define IFERR\_ABORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, TRUE))

#define IFERR\_REPORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, FALSE))

static int report\_error(char \*file, int line, char \*call, int status,

logical exit\_on\_error);

static void ECHO(char \*format, ...);

static void GTAC\_free(void \*what);

static int GTAC\_find\_rev(char \*item\_id, char \*rev\_id, tag\_t \*rev);

static void do\_it(void)

{

tag\_t end\_item\_rev = NULLTAG;

IFERR\_ABORT(ITEM\_find\_rev("EndItem", "A", &end\_item\_rev ));

if (end\_item\_rev == NULLTAG)

{

ECHO("End Item Revision not found!\n");

exit (0);

}

tag\_t assy\_rev = NULLTAG;

IFERR\_ABORT(ITEM\_find\_rev("Assy", "A", &assy\_rev));

if (end\_item\_rev == NULLTAG)

{

ECHO("Assy Revision not found!\n");

exit (0);

}

int n\_bvrs = 0;

tag\_t \*bvrs = NULL;

IFERR\_ABORT(ITEM\_rev\_list\_bom\_view\_revs(assy\_rev, &n\_bvrs, &bvrs));

if (n\_bvrs == 0)

{

ECHO("Assy Revision BVR not found!\n");

exit (0);

}

// only be one bvr for this test case

tag\_t assy\_bvr = bvrs[0];

if (bvrs) MEM\_free(bvrs);

tag\_t window = NULLTAG;

IFERR\_REPORT(BOM\_create\_window (&window));

tag\_t rule = NULLTAG;

IFERR\_REPORT(CFM\_find("Latest Working", &rule));

IFERR\_REPORT(BOM\_set\_window\_config\_rule(window, rule));

IFERR\_REPORT(BOM\_set\_window\_pack\_all(window, TRUE));

tag\_t top\_line = NULLTAG;

IFERR\_REPORT(BOM\_set\_window\_top\_line(window, NULLTAG, assy\_rev, NULLTAG,

&top\_line));

int n\_children = 0;

tag\_t \*children = NULL;

IFERR\_REPORT(BOM\_line\_ask\_child\_lines(top\_line, &n\_children, &children));

// being lazy, just using first child line

tag\_t **bom\_line** = children[0];

if (children) MEM\_free(children);

int bl\_occurrence = 0;

IFERR\_REPORT(BOM\_line\_look\_up\_attribute( "bl\_occurrence", &bl\_occurrence));

tag\_t bl\_occ = NULLTAG;

IFERR\_REPORT(BOM\_line\_ask\_attribute\_tag(**bom\_line**, bl\_occurrence, &bl\_occ));

IFERR\_REPORT(AOM\_refresh(assy\_bvr, TRUE));

IFERR\_REPORT(AOM\_refresh(bl\_occ, TRUE));

tag\_t occ\_eff = NULLTAG;

IFERR\_REPORT(PS\_occ\_eff\_create (assy\_bvr, bl\_occ, &occ\_eff));

IFERR\_REPORT(AOM\_save(occ\_eff));

IFERR\_REPORT(AOM\_save(bl\_occ));

IFERR\_REPORT(AOM\_save(assy\_bvr));

IFERR\_REPORT(PS\_occ\_eff\_set\_id(assy\_bvr, bl\_occ, occ\_eff, "Effectivity1"));

IFERR\_REPORT(PS\_occ\_eff\_set\_enditemrev(assy\_bvr, bl\_occ, occ\_eff,

end\_item\_rev));

IFERR\_REPORT(PS\_occ\_eff\_set\_unit\_range(assy\_bvr, bl\_occ, occ\_eff,

"1-UP", true));

IFERR\_REPORT(AOM\_save(occ\_eff));

IFERR\_REPORT(AOM\_unlock(occ\_eff));

IFERR\_REPORT(AOM\_unload(occ\_eff));

IFERR\_REPORT(AOM\_save(bl\_occ));

IFERR\_REPORT(AOM\_unlock(bl\_occ));

IFERR\_REPORT(AOM\_unload(bl\_occ));

IFERR\_REPORT(AOM\_save(assy\_bvr));

IFERR\_REPORT(AOM\_unlock(assy\_bvr));

IFERR\_REPORT(AOM\_unload(assy\_bvr));

IFERR\_REPORT(BOM\_save\_window(window));

IFERR\_REPORT(BOM\_close\_window(window));

}

int ITK\_user\_main(int argc, char\* argv[])

{

IFERR\_REPORT(ITK\_initialize\_text\_services(ITK\_BATCH\_TEXT\_MODE));

IFERR\_ABORT(ITK\_auto\_login());

IFERR\_REPORT(ITK\_set\_journalling(TRUE));

do\_it();

IFERR\_REPORT(ITK\_exit\_module(FALSE));

return ITK\_ok;

}

static int report\_error(char \*file, int line, char \*call, int status,

logical exit\_on\_error)

{

if (status != ITK\_ok)

{

int

n\_errors = 0;

const int

\*severities = NULL,

\*statuses = NULL;

const char

\*\*messages;

EMH\_ask\_errors(&n\_errors, &severities, &statuses, &messages);

if (n\_errors > 0)

{

ECHO("\n%s\n", messages[n\_errors-1]);

EMH\_clear\_errors();

}

else

{

char \*error\_message\_string;

EMH\_get\_error\_string (NULLTAG, status, &error\_message\_string);

ECHO("\n%s\n", error\_message\_string);

}

ECHO("error %d at line %d in %s\n", status, line, file);

ECHO("%s\n", call);

if (exit\_on\_error)

{

ECHO("%s", "Exiting program!\n");

exit (status);

}

}

return status;

}

static void ECHO(char \*format, ...)

{

char msg[1000];

va\_list args;

va\_start(args, format);

vsprintf(msg, format, args);

va\_end(args);

printf(msg);

TC\_write\_syslog(msg);

}

static void GTAC\_free(void \*what)

{

if (what != NULL)

{

MEM\_free(what);

what = NULL;

}

}

static int GTAC\_find\_rev(char \*item\_id, char \*rev\_id, tag\_t \*rev)

{

int

n = 0;

tag\_t

\*revs;

const char

\*names[1] = { "item\_id" },

\*values[1] = { item\_id };

IFERR\_REPORT(ITEM\_find\_item\_revs\_by\_key\_attributes(1, names, values,

rev\_id,

&n, &revs));

if (n > 0) \*rev = revs[0];

if (revs) MEM\_free(revs);

return 0;

}

#ifdef \_\_cplusplus

}

#endif

#  No **ITK** function found to remove Child Module Constraints

 Symptom

 ---------------

No **ITK** function found to remove Child Module Constraints in Modular Variant

 Hardware/Software Configuration

 Platform: INTEL

OS: WINDOW

OS Version: 2008\_R2

Family: TEAMCENTER

Application: CUSTOMIZATION

Function: **ITK**

Subfunction: ALL

Release: V9.1.2.4

 Solution

 Child Module Constraints saved as MVL you can find 2 **ITK** functions to ask or

set MLV data for a bomline.

. BOM\_line\_ask\_mvl to ask the MLV value

. BOM\_line\_set\_mvl to set the MLV value

MLV value is a string and you can manipulate by your self.

you can delete MVL entry by setting an empty string.

IFERR\_ABORT(BOM\_line\_ask\_mvl ( top\_line, &mvl));

printf("\t mvl: %s\n",mvl );

MEM\_free(mvl);

IFERR\_ABORT(BOM\_line\_set\_mvl ( top\_line, ""));

printf("\n mvl deleted \n" );

IFERR\_ABORT(BOM\_line\_ask\_mvl ( top\_line, &mvl));

printf("\t mvl: %s\n",mvl );

MEM\_free(mvl);

#  Sample **ITK** function : write plmxml file

#include <bom/bom.h>

#include <cfm/cfm.h>

static void write\_plmxml\_file(tag\_t item\_rev)

{

int ifail = **ITK**\_ok;

tag\_t window = NULLTAG;

IFERR\_ABORT(BOM\_create\_window (&window));

tag\_t rule = NULLTAG;

IFERR\_ABORT(CFM\_find("Latest Working--", &rule));

IFERR\_ABORT(BOM\_set\_window\_config\_rule(window, rule));

IFERR\_ABORT(BOM\_set\_window\_pack\_all(window, TRUE));

tag\_t top\_line = NULLTAG;

IFERR\_ABORT(BOM\_set\_window\_top\_line(window, NULLTAG, item\_rev, NULLTAG, &top\_line));

int n\_selected = 0;

tag\_t \*selected = NULL;

IFERR\_ABORT(**BOM\_line**\_ask\_child\_lines(top\_line, &n\_selected, &selected));

BOM\_writer\_output\* output = 0;

ifail = BOM\_writer\_new\_output\_file( &output );

output->file.filehandle = fopen("W:\\the\_file.xml", "r");

output->common.object = MEM\_string\_copy("W:\\the\_file.xml");

BOM\_writer\_format\* format = 0;

IFERR\_ABORT(BOM\_writer\_new\_format\_plmxml(&format));

format->plmxml.builder\_name = MEM\_string\_copy("AbsoluteOccurrences");

format->plmxml.transform\_type = TransformType\_AbsOcc;

format->plmxml.transfer\_mode = MEM\_string\_copy("ConfiguredDataFilesExportDefault");

BOM\_writer\_traversal\* traversal = 0;

IFERR\_ABORT(BOM\_writer\_new\_traversal(&traversal));

traversal->selected\_count = n\_selected;

traversal->selected\_lines = selected;

traversal->no\_descendants = false;

traversal->transient\_unpack = true;

IFERR\_ABORT(BOM\_writer\_write\_bomwindow(window, output, format, traversal));

MEM\_free(selected);

IFERR\_REPORT(BOM\_close\_window(window));

}

#  Sample **ITK** program : create product structure

/\*HEAD CREATE\_PRODUCT\_STRUCTURE CCC **ITK** \*/

#ifdef \_\_cplusplus

extern "C" {

#endif

#include <stdarg.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <tccore/aom.h>

#include <bom/bom.h>

#include <tc/emh.h>

#include <tc/folder.h>

#include <tc/tc.h>

#include <sa/tcfile.h>

#include <tccore/item.h>

#include <**itk**/mem.h>

#include <ps/ps.h>

#include <sa/user.h>

#include <tccore/tctype.h>

#define IFERR\_ABORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, TRUE))

#define IFERR\_REPORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, FALSE))

static void ECHO(char \*format, ...)

{

char msg[1000];

va\_list args;

va\_start(args, format);

vsprintf(msg, format, args);

va\_end(args);

printf(msg);

TC\_write\_syslog(msg);

}

static int report\_error(char \*file, int line, char \*call, int status,

logical exit\_on\_error)

{

if (status != ITK\_ok)

{

int

n\_errors = 0;

const int

\*severities = NULL,

\*statuses = NULL;

const char

\*\*messages;

EMH\_ask\_errors(&n\_errors, &severities, &statuses, &messages);

if (n\_errors > 0)

{

ECHO("\n%s\n", messages[n\_errors-1]);

EMH\_clear\_errors();

}

else

{

char \*error\_message\_string;

EMH\_get\_error\_string (NULLTAG, status, &error\_message\_string);

ECHO("\n%s\n", error\_message\_string);

}

ECHO("error %d at line %d in %s\n", status, line, file);

ECHO("%s\n", call);

if (exit\_on\_error)

{

ECHO("%s", "Exiting program!\n");

exit (status);

}

}

return status;

}

void insert\_object\_into\_folder(tag\_t folder, tag\_t object, int pos)

{

IFERR\_REPORT(AOM\_refresh( folder, TRUE));

IFERR\_REPORT(FL\_insert(folder, object, pos));

IFERR\_REPORT(AOM\_save(folder));

IFERR\_REPORT(AOM\_refresh( folder, FALSE));

}

void create\_folder\_in\_home\_folder(char \*folder\_name, tag\_t \*folder)

{

char

\*user\_name\_string = NULL;

tag\_t

user = NULLTAG,

home\_folder = NULLTAG;

IFERR\_REPORT(POM\_get\_user(&user\_name\_string, &user));

IFERR\_REPORT(SA\_ask\_user\_home\_folder(user, &home\_folder));

printf(" Creating Folder: Home-> %s\n", folder\_name);

IFERR\_REPORT(FL\_create(folder\_name, "", folder));

IFERR\_REPORT(AOM\_save(\*folder));

IFERR\_REPORT(FL\_insert(home\_folder, \*folder, -1));

IFERR\_REPORT(AOM\_save(home\_folder));

IFERR\_REPORT(AOM\_refresh( home\_folder, FALSE));

MEM\_free(user\_name\_string);

}

static void case\_insensitive\_find\_type(char \*input\_name, char \*\*item\_type, tag\_t \*type\_tag)

{

int number\_of\_types = 0;

tag\_t \*type\_tags = NULL;

IFERR\_REPORT(TCTYPE\_extent(&number\_of\_types, &type\_tags));

logical is\_valid\_type\_name = FALSE;

char type\_name[TCTYPE\_name\_size\_c+1] = "";

for(int ii = 0; ii < number\_of\_types; ii++)

{

IFERR\_REPORT(TCTYPE\_ask\_name(type\_tags[ii], type\_name));

if(tc\_strcasecmp(type\_name, input\_name) == 0)

{

IFERR\_REPORT(TCTYPE\_find\_type(type\_name, "", type\_tag));

is\_valid\_type\_name = TRUE;

\*item\_type = (char \*)MEM\_alloc(sizeof(char) \* (strlen(type\_name) +1 ) );

strcpy(\*item\_type, type\_name);

break;

}

}

MEM\_free(type\_tags);

}

void usage ()

{

printf ("\n\n USAGE: \n");

printf ("\n\n create\_product\_structure -i=<item id> -t=<type> -n=<number of child lines>\n");

return;

}

static void do\_it(void)

{

int

n\_children = 0,

ii = 0;

tag\_t

folder = NULLTAG,

assy\_item = NULLTAG,

assy\_rev = NULLTAG,

window = NULLTAG,

top\_line = NULLTAG,

bv = NULLTAG,

bvr = NULLTAG,

item = NULLTAG,

rev = NULLTAG,

child\_line = NULLTAG;

char

assy\_name[ITEM\_id\_size\_c + 1] = "";

if( (ITK\_ask\_cli\_argument ("-h") != NULL) ||

(ITK\_ask\_cli\_argument("-i=") == NULL) ||

(ITK\_ask\_cli\_argument("-t=") == NULL) ||

(ITK\_ask\_cli\_argument("-n=") == NULL) )

{

usage();

exit(0);

}

char \*item\_id = ITK\_ask\_cli\_argument("-i=");

char \*input\_name = ITK\_ask\_cli\_argument("-t=");

char \*children = ITK\_ask\_cli\_argument("-n=");

char \*item\_type = NULL;

tag\_t type\_tag = NULLTAG;

IFERR\_REPORT(TCTYPE\_find\_type(input\_name, "", &type\_tag));

if (type\_tag != NULLTAG)

{

item\_type = (char \*)MEM\_alloc(sizeof(char) \* (strlen(input\_name) +1 ) );

strcpy(item\_type, input\_name);

}

else

{

int number\_of\_types = 0;

tag\_t \*type\_tags = NULL;

IFERR\_REPORT(TCTYPE\_extent(&number\_of\_types, &type\_tags));

char type\_name[TCTYPE\_name\_size\_c+1] = "";

for(int ii = 0; ii < number\_of\_types; ii++)

{

IFERR\_REPORT(TCTYPE\_ask\_name(type\_tags[ii], type\_name));

if(tc\_strcasecmp(type\_name, input\_name) == 0)

{

IFERR\_REPORT(TCTYPE\_find\_type(type\_name, "", &type\_tag));

item\_type = (char \*)MEM\_alloc(sizeof(char) \* (strlen(type\_name) +1 ) );

strcpy(item\_type, type\_name);

break;

}

}

MEM\_free(type\_tags);

}

if (type\_tag == NULLTAG)

{

printf("\n\n Busineess Object type \"%s\" NOT found! \n\n", input\_name);

exit (0);

}

n\_children = atoi(children);

strcpy(assy\_name, item\_id);

printf(" Creating Item: %s \n", item\_id);

IFERR\_ABORT(ITEM\_create\_item(item\_id, "Top Assembly", item\_type, "A", &assy\_item, &assy\_rev));

IFERR\_REPORT(AOM\_save (assy\_item));

create\_folder\_in\_home\_folder(item\_id, &folder);

insert\_object\_into\_folder(folder, assy\_item, -1);

IFERR\_REPORT(PS\_create\_bom\_view (NULLTAG, "", "", assy\_item, &bv));

IFERR\_REPORT(AOM\_save (bv));

IFERR\_REPORT(PS\_create\_bvr (bv, "", "", false, assy\_rev, &bvr));

IFERR\_REPORT(AOM\_save (bvr));

IFERR\_REPORT(AOM\_save (assy\_item));

for (ii = 0; ii < n\_children; ii++)

{

if (ii == 0)

{

IFERR\_REPORT(BOM\_create\_window (&window));

IFERR\_REPORT(BOM\_set\_window\_top\_line(window, assy\_item, assy\_rev,

NULLTAG, &top\_line));

}

sprintf(item\_id, "%s\_%d", assy\_name, ii + 1);

printf(" Creating Item: %s\n", item\_id);

IFERR\_REPORT(ITEM\_create\_item(item\_id, "Component", item\_type, "A",

&item, &rev) );

IFERR\_REPORT(AOM\_save(item));

insert\_object\_into\_folder(folder, item, 999);

IFERR\_REPORT(BOM\_line\_add(top\_line, item, rev, NULLTAG, &child\_line));

IFERR\_REPORT(BOM\_save\_window(window));

}

IFERR\_REPORT(BOM\_close\_window(window));

}

int ITK\_user\_main(int argc, char\* argv[])

{

int

status = 0;

ITK\_initialize\_text\_services( ITK\_BATCH\_TEXT\_MODE );

status = ITK\_auto\_login();

if ( (status != ITK\_ok)) printf("\nLogin Failed!\n\n");

else

{

ITK\_set\_journalling(TRUE);

do\_it();

}

ITK\_exit\_module(TRUE);

return status;

}

#ifdef \_\_cplusplus

}

#endif

#  Sample **ITK** program : set item revision master form attribute based on bom attributes

/\*HEAD SET\_ITEM\_REVISION\_MASTER\_FORM\_ATTRIBUTE\_BASED\_ON\_BOM\_ATTRIBUTES CCC **ITK** \*/

#include<stdlib.h>

#include <bom/bom.h>

#include <cfm/cfm.h>

#include <tc/tc.h>

#include <tccore/tctype.h>

#include <tccore/item.h>

#include <ps/ps\_errors.h>

#include <property/prop.h>

#include <pom/pom/pom\_tokens.h>

#define EXIT\_FAILURE 1

#define ERROR\_CHECK(x) { \

int stat; \

char \*err\_string; \

if( (stat = (x)) != ITK\_ok) \

{ \

EMH\_get\_error\_string (NULLTAG, stat, &err\_string); \

printf ("ERROR: %d ERROR MSG: %s.\n", stat, err\_string); \

printf ("FUNCTION: %s\nFILE: %s LINE: %d\n",#x, \_\_FILE\_\_, \_\_LINE\_\_); \

if(err\_string) MEM\_free(err\_string); \

exit (EXIT\_FAILURE); \

} \

}

static char total[256] = "";

static void set\_item\_revisions\_master\_form\_property(tag\_t item\_revision,

char \*property\_name, char \*property\_value)

{

int

n\_secondary\_objects = 0;

tag\_t

relation = NULLTAG,

\*secondary\_objects = NULL,

irm\_form = NULLTAG;

ERROR\_CHECK(GRM\_find\_relation\_type("IMAN\_master\_form", &relation));

ERROR\_CHECK(GRM\_list\_secondary\_objects\_only(item\_revision, relation,

&n\_secondary\_objects, &secondary\_objects) );

/\* should always be just one \*/

irm\_form = secondary\_objects[0];

if (secondary\_objects) MEM\_free(secondary\_objects);

ERROR\_CHECK(AOM\_refresh(irm\_form, TRUE));

ERROR\_CHECK(AOM\_set\_value\_string(irm\_form, property\_name, property\_value));

ERROR\_CHECK(AOM\_save(irm\_form));

ERROR\_CHECK(AOM\_unload(irm\_form));

}

static tag\_t ask\_item\_revisions\_master\_form(tag\_t item\_revision)

{

int

n\_secondary\_objects = 0;

tag\_t

relation = NULLTAG,

\*secondary\_objects = NULL,

item\_revision\_master\_form = NULLTAG;

ERROR\_CHECK(GRM\_find\_relation\_type("IMAN\_master\_form", &relation));

ERROR\_CHECK(GRM\_list\_secondary\_objects\_only(item\_revision, relation,

&n\_secondary\_objects, &secondary\_objects));

/\* should always be just one \*/

item\_revision\_master\_form = secondary\_objects[0];

if (secondary\_objects) MEM\_free(secondary\_objects);

return item\_revision\_master\_form;

}

static tag\_t ask\_item\_revision\_from\_bom\_line(tag\_t **bom\_line**)

{

tag\_t

item\_revision = NULLTAG;

char

\*item\_id = NULL,

\*rev\_id = NULL;

ERROR\_CHECK(AOM\_ask\_value\_string(**bom\_line**, "bl\_item\_item\_id", &item\_id ));

ERROR\_CHECK(AOM\_ask\_value\_string(**bom\_line**, "bl\_rev\_item\_revision\_id",

&rev\_id));

ERROR\_CHECK(ITEM\_find\_rev(item\_id, rev\_id, &item\_revision));

if (item\_id) MEM\_free(item\_id);

if (rev\_id) MEM\_free(rev\_id);

return item\_revision;

}

static void traverse\_product\_structure(tag\_t line, char \*field\_name)

{

int

ii, kk,

count;

tag\_t

\*children = NULL,

item\_revision = NULLTAG,

irm\_form = NULLTAG;

char

\*value = NULL;

item\_revision = ask\_item\_revision\_from\_bom\_line(line);

irm\_form = ask\_item\_revisions\_master\_form(item\_revision);

ERROR\_CHECK(AOM\_ask\_value\_string(irm\_form, field\_name, &value));

strcat(total, value);

ERROR\_CHECK(BOM\_line\_ask\_child\_lines(line, &count, &children));

for (ii = 0; ii < count; ii++)

traverse\_product\_structure(children[ii], field\_name);

if (value) MEM\_free(value);

if (children) MEM\_free(children);

}

static void send\_item\_revision\_to\_pse(tag\_t item\_revision, char \*field\_name)

{

tag\_t

rule = NULLTAG,

window = NULLTAG,

bvr = NULLTAG,

top\_line = NULLTAG;

ERROR\_CHECK(BOM\_create\_window (&window));

ERROR\_CHECK(CFM\_find("Latest Working", &rule));

ERROR\_CHECK(BOM\_set\_window\_config\_rule(window, rule));

ERROR\_CHECK(BOM\_set\_window\_pack\_all(window, TRUE));

ERROR\_CHECK(BOM\_set\_window\_top\_line(window, NULLTAG, item\_revision, bvr,

&top\_line));

traverse\_product\_structure(top\_line, field\_name);

ERROR\_CHECK(BOM\_close\_window(window));

}

static void do\_it(void)

{

tag\_t

item = NULLTAG,

item\_revision = NULLTAG;

char

\*item\_id = ITK\_ask\_cli\_argument("-i="),

\*field\_name = ITK\_ask\_cli\_argument("-f=");

ERROR\_CHECK(ITEM\_find\_item(item\_id, &item));

if (!item)

{

printf("\tItem %s not found!\n", item\_id);

exit(EXIT\_FAILURE);

}

ERROR\_CHECK(ITEM\_ask\_latest\_rev(item, &item\_revision));

send\_item\_revision\_to\_pse(item\_revision, field\_name);

set\_item\_revisions\_master\_form\_property(item\_revision, field\_name, total);

}

int ITK\_user\_main(int argc, char\* argv[])

{

int

status = 0;

char

\*message;

ERROR\_CHECK(ITK\_initialize\_text\_services( 0 ));

status = ITK\_auto\_login();

if ( (status != ITK\_ok)) printf("iMAN login NOT successful.\n");

else

{

printf("iMAN login successful.\n\n");

ERROR\_CHECK(ITK\_set\_journalling(TRUE));

do\_it();

}

ITK\_exit\_module(TRUE);

return status;

}

#  Sample **ITK** program : remove all bomline substitutes

/\*HEAD REMOVE\_ALL\_BOMLINE\_SUBSTITUTES CCC **ITK** \*/

#include <stdlib.h>

#include <tccore/aom.h>

#include <pom/pom/pom.h>

#include <tc/emh.h>

#include <ict/ict\_userservice.h>

#include <tc/tc.h>

#include <tccore/tctype.h>

#include <imanf

#include <tccore/item.h>

#include <ss/ss\_errors.h>

#define ECHO(X) printf X; TC\_write\_syslog X

#define IFERR\_ABORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, TRUE))

#define IFERR\_REPORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, FALSE))

#define IFERR\_RETURN(X) if (IFERR\_REPORT(X)) return

static int report\_error(char \*file, int line, char \*call, int status,

logical exit\_on\_error)

{

if (status != ITK\_ok)

{

int

n\_errors = 0,

\*severities = NULL,

\*statuses = NULL;

char

\*\*messages;

EMH\_ask\_errors( &n\_errors, &severities, &statuses, &messages );

if (n\_errors > 0)

{

ECHO(("\n%s\n", messages[n\_errors-1]));

EMH\_clear\_errors();

}

else

{

char \*error\_message\_string;

EMH\_get\_error\_string (NULLTAG, status, &error\_message\_string);

ECHO(("\n%s\n", error\_message\_string));

}

ECHO(("error %d at line %d in %s\n", status, line, file));

ECHO(("%s\n", call));

if (exit\_on\_error)

{

ECHO(("\nExiting program!\n"));

exit (status);

}

}

return status;

}

static void GTAC\_free(void \*what)

{

if (what != NULL)

{

MEM\_free(what);

what = NULL;

}

}

static void do\_it(void)

{

logical

verdict = FALSE;

int

ii = 0,

n\_children;

tag\_t

rev\_tag = NULLTAG,

window = NULLTAG,

top\_line = NULLTAG,

\*children = NULL;

char

\*title = NULL;

IFERR\_REPORT(ITEM\_find\_rev ("EndItem", "00", &rev\_tag));

ECHO(("rev\_tag: %u\n", rev\_tag));

IFERR\_REPORT(BOM\_create\_window (&window));

IFERR\_REPORT(BOM\_set\_window\_top\_line(window, NULLTAG, rev\_tag, NULLTAG, &top\_line));

IFERR\_REPORT(BOM\_line\_ask\_child\_lines(top\_line, &n\_children, &children));

for (ii = 0; ii < n\_children; ii++)

{

IFERR\_REPORT(AOM\_ask\_value\_string(children[ii], "bl\_formatted\_title", &title));

ECHO(("bl\_formatted\_title: %s \n", title));

IFERR\_REPORT(BOM\_line\_ask\_is\_substitute(children[ii], &verdict));

if (verdict) IFERR\_REPORT(BOM\_line\_cut(children[ii]));

GTAC\_free(title);

}

GTAC\_free(children);

IFERR\_REPORT(BOM\_save\_window(window));

}

int ITK\_user\_main(int argc, char\* argv[])

{

IFERR\_REPORT(ITK\_initialize\_text\_services(ITK\_BATCH\_TEXT\_MODE));

IFERR\_ABORT(ITK\_auto\_login());

IFERR\_REPORT(ITK\_set\_journalling(TRUE));

do\_it();

IFERR\_REPORT(ITK\_exit\_module(FALSE));

return ITK\_ok;

}

#  Sample **ITK** program : print custom bom view

/\*HEAD PRINT\_CUSTOM\_BOM\_VIEW CCC **ITK** \*/

#include<stdlib.h>

#include <bom/bom.h>

#include <cfm/cfm.h>

#include <tc/tc.h>

#include <tccore/item.h>

#include <ps/ps\_errors.h>

#define EXIT\_FAILURE 1

#define ITK\_CALL(x) { \

int stat; \

char \*err\_string; \

if( (stat = (x)) != ITK\_ok) \

{ \

EMH\_get\_error\_string (NULLTAG, stat, &err\_string); \

printf ("ERROR: %d ERROR MSG: %s.\n", stat, err\_string); \

printf ("FUNCTION: %s\nFILE: %s LINE: %d\n",#x, \_\_FILE\_\_, \_\_LINE\_\_); \

if(err\_string) MEM\_free(err\_string); \

exit (EXIT\_FAILURE); \

} \

}

static int name\_att, qty\_att, id\_att, desc\_att;

static void look\_up\_bom\_attributes(void)

{

ITK\_CALL(BOM\_line\_look\_up\_attribute(bomAttr\_lineName, &name\_att));

ITK\_CALL(BOM\_line\_look\_up\_attribute(bomAttr\_occQty, &qty\_att));

ITK\_CALL(BOM\_line\_look\_up\_attribute(bomAttr\_itemId, &id\_att));

ITK\_CALL(BOM\_line\_look\_up\_attribute(bomAttr\_itemDesc, &desc\_att));

}

static void print\_bom (tag\_t line, int indention)

{

int

ii,

count;

char

\*name,

\*qty,

\*id,

\*desc;

tag\_t

\*children;

if (indention == 0) printf(" QTY ITEM ID DESCRIPTION\n");

indention++;

ITK\_CALL(BOM\_line\_ask\_attribute\_string(line, name\_att, &name));

ITK\_CALL(BOM\_line\_ask\_attribute\_string(line, qty\_att, &qty));

if (strcmp(qty,"") == 0) strcpy(qty, "");

ITK\_CALL(BOM\_line\_ask\_attribute\_string(line, id\_att, &id ));

ITK\_CALL(BOM\_line\_ask\_attribute\_string(line, desc\_att, &desc));

for (ii = 0; ii < indention; ii++) printf (" ");

printf("%s\t%-20s\t%-20s\n", qty, id, desc);

ITK\_CALL(BOM\_line\_ask\_child\_lines(line, &count, &children));

for (ii = 0; ii < count; ii++) print\_bom (children[ii], indention);

if (name) MEM\_free(name);

if (qty) MEM\_free(qty);

if (id) MEM\_free(id);

if (desc) MEM\_free(desc);

if (children) MEM\_free(children);

}

static void create\_bom\_window(tag\_t item, char \*view\_name, tag\_t \*top\_line)

{

int

num\_of\_boms,

ii;

char

\*type\_name;

tag\_t

rule,

window,

bom\_view = NULLTAG,

\*bvs,

view\_type;

ITK\_CALL(BOM\_create\_window (&window));

ITK\_CALL(CFM\_find("Latest Working", &rule));

ITK\_CALL(BOM\_set\_window\_config\_rule(window, rule));

ITK\_CALL(BOM\_set\_window\_pack\_all(window, TRUE));

ITK\_CALL(ITEM\_list\_bom\_views( item, &num\_of\_boms, &bvs));

for (ii = 0; ii < num\_of\_boms; ii++)

{

ITK\_CALL(PS\_ask\_bom\_view\_type(bvs[ii], &view\_type));

ITK\_CALL(PS\_ask\_view\_type\_name(view\_type, &type\_name));

if (!strcmp(type\_name, view\_name)) bom\_view = bvs[ii];

}

if (bom\_view == NULLTAG)

printf("BOMview: \"%s\" not found!\nUsing default BOMview.\n\n", view\_name);

ITK\_CALL(BOM\_set\_window\_top\_line(window, item, NULLTAG, bom\_view, top\_line));

}

static void do\_it(void)

{

int

indention = 0;

char

\*item\_name = ITK\_ask\_cli\_argument("-i="),

\*view\_name = ITK\_ask\_cli\_argument("-v=");

tag\_t

item,

window,

top\_line;

ITK\_CALL(ITEM\_find\_item (item\_name, &item));

if (!item)

{

printf("\tItem %s does not exist!\n", item\_name);

exit(EXIT\_FAILURE);

}

create\_bom\_window(item, view\_name, &top\_line);

look\_up\_bom\_attributes();

print\_bom(top\_line, indention);

}

int ITK\_user\_main(int argc, char\* argv[])

{

int

status = 0;

char

\*message;

ITK\_CALL(ITK\_initialize\_text\_services( 0 ));

status = ITK\_auto\_login();

if ( (status != ITK\_ok)) printf("iMAN login NOT successful.\n");

else

{

printf("iMAN login successful.\n\n");

ITK\_CALL(ITK\_set\_journalling(TRUE));

do\_it();

}

ITK\_exit\_module(TRUE);

return status;

}

#  Sample **ITK** function : ask item revision from bom line

static tag\_t ask\_item\_revision\_from\_bom\_line(tag\_t **bom\_line**)

{

tag\_t

item\_revision = NULLTAG;

char

\*item\_id = NULL,

\*rev\_id = NULL;

ERROR\_CHECK(AOM\_ask\_value\_string(**bom\_line**, "bl\_item\_item\_id", &item\_id ));

ERROR\_CHECK(AOM\_ask\_value\_string(**bom\_line**, "bl\_rev\_item\_revision\_id",

&rev\_id));

ERROR\_CHECK(ITEM\_find\_rev(item\_id, rev\_id, &item\_revision));

if (item\_id) MEM\_free(item\_id);

if (rev\_id) MEM\_free(rev\_id);

return item\_revision;

}

#  Sample **ITK** program : remove all bomline substitutes

/\*HEAD REMOVE\_ALL\_BOMLINE\_SUBSTITUTES CCC **ITK** \*/

#include <stdlib.h>

#include <tccore/aom.h>

#include <pom/pom/pom.h>

#include <tc/emh.h>

#include <ict/ict\_userservice.h>

#include <tc/tc.h>

#include <tccore/tctype.h>

#include <imanf

#include <tccore/item.h>

#include <ss/ss\_errors.h>

#define ECHO(X) printf X; TC\_write\_syslog X

#define IFERR\_ABORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, TRUE))

#define IFERR\_REPORT(X) (report\_error( \_\_FILE\_\_, \_\_LINE\_\_, #X, X, FALSE))

#define IFERR\_RETURN(X) if (IFERR\_REPORT(X)) return

static int report\_error(char \*file, int line, char \*call, int status,

logical exit\_on\_error)

{

if (status != ITK\_ok)

{

int

n\_errors = 0,

\*severities = NULL,

\*statuses = NULL;

char

\*\*messages;

EMH\_ask\_errors( &n\_errors, &severities, &statuses, &messages );

if (n\_errors > 0)

{

ECHO(("\n%s\n", messages[n\_errors-1]));

EMH\_clear\_errors();

}

else

{

char \*error\_message\_string;

EMH\_get\_error\_string (NULLTAG, status, &error\_message\_string);

ECHO(("\n%s\n", error\_message\_string));

}

ECHO(("error %d at line %d in %s\n", status, line, file));

ECHO(("%s\n", call));

if (exit\_on\_error)

{

ECHO(("\nExiting program!\n"));

exit (status);

}

}

return status;

}

static void GTAC\_free(void \*what)

{

if (what != NULL)

{

MEM\_free(what);

what = NULL;

}

}

static void do\_it(void)

{

logical

verdict = FALSE;

int

ii = 0,

n\_children;

tag\_t

rev\_tag = NULLTAG,

window = NULLTAG,

top\_line = NULLTAG,

\*children = NULL;

char

\*title = NULL;

IFERR\_REPORT(ITEM\_find\_rev ("EndItem", "00", &rev\_tag));

ECHO(("rev\_tag: %u\n", rev\_tag));

IFERR\_REPORT(BOM\_create\_window (&window));

IFERR\_REPORT(BOM\_set\_window\_top\_line(window, NULLTAG, rev\_tag, NULLTAG, &top\_line));

IFERR\_REPORT(BOM\_line\_ask\_child\_lines(top\_line, &n\_children, &children));

for (ii = 0; ii < n\_children; ii++)

{

IFERR\_REPORT(AOM\_ask\_value\_string(children[ii], "bl\_formatted\_title", &title));

ECHO(("bl\_formatted\_title: %s \n", title));

IFERR\_REPORT(BOM\_line\_ask\_is\_substitute(children[ii], &verdict));

if (verdict) IFERR\_REPORT(BOM\_line\_cut(children[ii]));

GTAC\_free(title);

}

GTAC\_free(children);

IFERR\_REPORT(BOM\_save\_window(window));

}

int ITK\_user\_main(int argc, char\* argv[])

{

IFERR\_REPORT(ITK\_initialize\_text\_services(ITK\_BATCH\_TEXT\_MODE));

IFERR\_ABORT(ITK\_auto\_login());

IFERR\_REPORT(ITK\_set\_journalling(TRUE));

do\_it();

IFERR\_REPORT(ITK\_exit\_module(FALSE));

return ITK\_ok;

}

**´**

#  Revision Rule - **ITK**

 Hello everyone,

Following is a **ITK** code segment which loads a BVR, sets a revision rule and

tries to set the BVR as precise and save it. So the expected result is that

the next time I open the BVR, I get the same revisions for the children as

it is loaded now...

-------CODE STARTS HERE-------

BOM\_create\_window(&window);

BOM\_set\_window\_top\_line(window,top\_item\_tag,top\_rev\_tag,bv,&top\_line);

BOM\_ask\_window\_config\_rule(window,&rule);

CFM\_ITEM\_set\_config(rule,CFM\_latest\_status,FALSE,"Draft",-1,CFM\_null\_date,NULLTAG);

BOM\_line\_ask\_child\_lines(top\_line,&n\_children,&children);

BOM\_line\_look\_up\_attribute(bomAttr\_lineItemRevTag,&rev\_tag\_attr);

for(int inx = 0;inx < n\_children;inx++)

{

tag\_t item\_rev;

char rev\_id[ITEM\_id\_size\_c +1] = "\0";

BOM\_line\_ask\_attribute\_tag(children[inx],rev\_tag\_attr,&item\_rev);

ITEM\_ask\_rev\_id(item\_rev,rev\_id);

cout << "Rev Id " << rev\_id << "\n";

}

BOM\_line\_set\_precise(top\_line,TRUE);

AOM\_save(bvrs[0]);

BOM\_save\_window(window);

BOM\_close\_window(window);

-------CODE ENDS HERE-------

The result of this code:

The BVR is set as precise, but when I open it in the PSE the children are

not having the same revisions as shown by the code (Not the configured ones).

# Creation and management of product configurator objects through **ITK** customization

 We want to know if there are any **ITKs** available to create product configurator objects such as Configurator context, Family group, Option family, options and saved **variant** **rule** from outside of Teamcenter

 Hardware/Software Configuration

 Platform: INTL64

OS: SUSE

OS Version: SLED11S3

Product: TEAMCENTER

Application: CONFIGURATOR

Version: V10.1.7.1

Function: API

 Solution

 Administrator can use standard API for creating context and other configurator objects.  
   
**For Product context use ITK**: "ITEM\_create\_item"  
Example:  
ITEM\_create\_item ( "", "ModelFamilyWSO", "Cfg0ProductItem", "A", item1, item\_rev1 )  
   
**For Option Families, values etc**. use TCTYPE\_create\_object **ITK**.  
Example:  
TCTYPE\_ask\_type ( "Cfg0ProductModelFamily", wsoType )  
TCTYPE\_construct\_create\_input( wsoType, createInputTagWso )  
   
**@\* populate with values**  
AOM\_set\_value\_string( createInputTagWso, "cfg0ObjectId", "Engines")  
AOM\_set\_value\_string( createInputTagWso, "object\_name", "Engine Models")  
AOM\_set\_value\_logical( createInputTagWso, "cfg0IsDiscretionary", false)  
AOM\_set\_value\_logical( createInputTagWso , "cfg0HasFreeFormValues", false)  
AOM\_set\_value\_logical( createInputTagWso , "cfg0IsMultiselect", false)  
AOM\_set\_value\_tag( createInputTagWso, "cfg0OwningProductItem", item1 )  
AOM\_set\_value\_string( createInputTagWso, "cfg0FamilyNamespace", itemId )  
AOM\_set\_value\_string( createInputTagWso , "cfg0ValueDataType", CFG0CONFIGURATOR\_String\_Value\_Type )  
   
**@\* create object and save**  
TCTYPE\_create\_object( createInputTagWso, boTag )  
   
   
**To link an option family/value or group to a context you need to allocate them to context.**  
   
Example:  
@\* Create allocation record for family  
TCTYPE\_ask\_type ( "Cfg0Allocation", allocationType )  
TCTYPE\_construct\_create\_input( allocationType, allocationCreInput )  
AOM\_set\_value\_string( allocationCreInput, "cfg0ObjectId", "FamilyAllocation" )  
AOM\_set\_value\_string( allocationCreInput, "object\_name", "Family Allocation" )  
AOM\_set\_value\_tag( allocationCreInput, "cfg0ProductItem", itemWSO )  
AOM\_set\_value\_tag( allocationCreInput, "cfg0Target", optionFamilyWSOTag )  
TCTYPE\_create\_object( allocationCreInput, familyAllocationTag )  
AOM\_save( familyAllocationTag )  
AOM\_refresh( familyAllocationTag, 0 )  
   
 **Configurator Rules:**  
   
@\* Create Product Item  
TCTYPE\_ask\_type ( "Item", itemType )  
TCTYPE\_construct\_create\_input( itemType, itemCreateInputTag )  
   
AOM\_set\_value\_string( itemCreateInputTag, "object\_name", "product\_item" )  
TCTYPE\_create\_object( itemCreateInputTag, productItemTag )  
AOM\_save\_with\_extensions( productItemTag )  
   
   
@\* Create  Default Rule Object without providing required fields.  
TCTYPE\_ask\_type ( "Cfg0DefaultRule", defaultRuleType )  
TCTYPE\_construct\_create\_input( defaultRuleType, defaultRuleCreateInputTag )  
   
@\* Create  Default Rule Object by providing required attributes  
@\* and without providing any Thread input descriptor.  
AOM\_set\_value\_string( defaultRuleCreateInputTag, "object\_name", "Default-Rule")  
AOM\_set\_value\_tags( defaultRuleCreateInputTag, "cfg0ProductItems", 1 , { productItemTag } )  
   
@\* Create rule object and save. This creates thread object in back ground.  
TCTYPE\_create\_object( defaultRuleCreateInputTag, defaultRuleTag )  
AOM\_save( defaultRuleTag )  
AOM\_refresh( defaultRuleTag, 0 )

 Reference

# Sample Teamcenter C++ function : new root item from template using saved **variant** **rule**

#include <iostream>

#include <bom/bom.h>

#include <cfm/cfm.h>

#include <collabctx/collabctx.h>

#include <me/me.h>

#include <tc/tc.h>

#include <tc/tc\_startup.h>

#include <tccore/item.h>

#include <tccore/grm.h>

#include <tccore/grmtype.h>

#include <tccore/tctype.h>

#include <user\_exits/user\_exits.h>

#include <base\_utils/TcResultStatus.hxx>

#include <base\_utils/IFail.hxx>

using namespace std;

static int new\_root\_item\_from\_template\_using\_saved\_**variant\_rule**(tag\_t tRev, tag\_t tSVR)

{

int ifail = **ITK**\_ok;

ResultStatus status;

try

{

tag\_t tWindow = NULLTAG;

status = BOM\_create\_window (&tWindow);

tag\_t tTopLine = NULLTAG;

status = BOM\_set\_window\_top\_line(tWindow, NULLTAG, tRev, NULLTAG, &tTopLine);

tag\_t tRule = NULLTAG;

status = CFM\_find("Latest Working", &tRule);

status = BOM\_set\_window\_config\_rule(tWindow, tRule);

status = BOM\_window\_hide\_variants (tWindow);

status = BOM\_window\_apply\_full\_vrule(tWindow, tSVR);

tag\_t ccContext = NULLTAG;

status = COLLABCTX\_create\_config\_context\_based\_on\_ref( "", "CCName", "CCDesc", &tWindow, &ccContext);

tag\_t tNewWindow = NULLTAG;

status = BOM\_create\_window (&tNewWindow);

tag\_t tItem = NULLTAG;

status = ITEM\_ask\_item\_of\_rev(tRev, &tItem);

tag\_t tType =NULLTAG;

status = TCTYPE\_ask\_object\_type(tItem, &tType);

logical isModified = FALSE;

char\* itemId = NULL;

status = USER\_new\_item\_id(NULLTAG, tType, &isModified, &itemId);

cout << "\n itemId: "<< itemId << endl;

tag\_t tNewRev;

char ruleKey[] = "Product.Template.Cloning";

status = ME\_create\_product\_from\_template(itemId, "A", itemId, "", tRev, ccContext, tNewWindow, ruleKey, &tNewRev);

status = BOM\_close\_window(tWindow);

status = BOM\_close\_window(tNewWindow);

if(itemId) MEM\_free(itemId);

}

catch( const IFail &e )

{

ifail = e.ifail();

cout << "\n error " << ifail << endl;

cout << " "<< e.getMessage() << endl;

}

return ifail;

}

# Sample Teamcenter C++ function : new child item from template using saved **variant** **rule**

#include <iostream>

#include <bom/bom.h>

#include <cfm/cfm.h>

#include <collabctx/collabctx.h>

#include <me/me.h>

#include <tc/tc.h>

#include <tc/tc\_startup.h>

#include <tccore/item.h>

#include <tccore/grm.h>

#include <tccore/grmtype.h>

#include <tccore/tctype.h>

#include <user\_exits/user\_exits.h>

#include <base\_utils/TcResultStatus.hxx>

#include <base\_utils/IFail.hxx>

using namespace std;

static int new\_child\_item\_from\_template\_using\_saved\_**variant\_rule**(tag\_t tRev, tag\_t tSVR)

{

int ifail = **ITK**\_ok;

ResultStatus status;

try

{

tag\_t tWindow = NULLTAG;

status = BOM\_create\_window (&tWindow);

tag\_t tTopLine = NULLTAG;

status = BOM\_set\_window\_top\_line(tWindow, NULLTAG, tRev, NULLTAG, &tTopLine);

tag\_t tRule = NULLTAG;

status = CFM\_find("Latest Working", &tRule);

status = BOM\_set\_window\_config\_rule(tWindow, tRule);

status = BOM\_window\_hide\_variants (tWindow);

status = BOM\_window\_apply\_full\_vrule(tWindow, tSVR);

tag\_t ccContext = NULLTAG;

status = COLLABCTX\_create\_config\_context\_based\_on\_ref( "", "CCName", "CCDesc", &tWindow, &ccContext);

tag\_t tItem = NULLTAG;

status = ITEM\_ask\_item\_of\_rev(tRev, &tItem);

tag\_t tType =NULLTAG;

status = TCTYPE\_ask\_object\_type(tItem, &tType);

logical isModified = FALSE;

char\* itemId = NULL;

status = USER\_new\_item\_id(NULLTAG, tType, &isModified, &itemId);

cout << "\n itemId: "<< itemId << endl;

tag\_t tNewRev;

char ruleKey[] = "Product.Template.Cloning";

status = ME\_create\_product\_from\_template(itemId, "A", itemId, "", tRev, ccContext, tWindow, ruleKey, &tNewRev);

status = BOM\_close\_window(tWindow);

if(itemId) MEM\_free(itemId);

}

catch( const IFail &e )

{

ifail = e.ifail();

cout << "\n error " << ifail << endl;

cout << " "<< e.getMessage() << endl;

}

return ifail;

}

# Sample **ITK** function : create saved **variant** **rule**

#include <tccore/grm.h>

#include <tccore/grmtype.h>

#include <bom/bom.h>

#include <ps/vrule.h>

static void create\_saved\_**variant\_rule**(tag\_t window, tag\_t rev\_tag, char \*target\_value)

{

int ifail = **ITK**\_ok;

tag\_t vrule\_tag = NULLTAG;

ifail = BOM\_window\_ask\_**variant\_rule**(window, &vrule\_tag);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

int n\_options = 0;

tag\_t \*options = NULL;

tag\_t \*option\_revs = NULL;

ifail = BOM\_**variant\_rule**\_ask\_options(vrule\_tag, &n\_options, &options , &option\_revs);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

/\* assuming only one \*/

tag\_t opt\_tag = options[0];

tag\_t opt\_rev\_tag = option\_revs[0];

if(options) MEM\_free(options);

if(option\_revs) MEM\_free(option\_revs);

int n\_values = 0;

int \*index = NULL;

ifail = BOM\_list\_option\_rev\_values(opt\_rev\_tag, &n\_values, &index);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

for(int ii = 0; ii < n\_values; ii++)

{

char \*value = NULL;

ifail = BOM\_ask\_option\_rev\_value(opt\_rev\_tag, index[ii], &value);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

if(strcmp(target\_value, value) == 0)

{

ifail = BOM\_**variant\_rule**\_set\_option\_values(vrule\_tag , opt\_tag, 1, &index[ii]);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

tag\_t saved\_vrule\_tag = NULLTAG;

ifail = VRULE\_create\_from\_**variant\_rule**("MyVarRule", "", vrule\_tag, 1, &opt\_tag, &saved\_vrule\_tag);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

tag\_t relation\_type\_tag = NULLTAG;

ifail = GRM\_find\_relation\_type( "IMAN\_specification", &relation\_type\_tag);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

tag\_t relation\_tag = NULLTAG;

ifail = GRM\_create\_relation(rev\_tag, saved\_vrule\_tag , relation\_type\_tag, NULLTAG, &relation\_tag);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

ifail = GRM\_save\_relation(relation\_tag);

if (ifail != **ITK**\_ok) { /\* your error logic here \*/ }

}

if(value) MEM\_free(value);

}

if(index) MEM\_free(index);

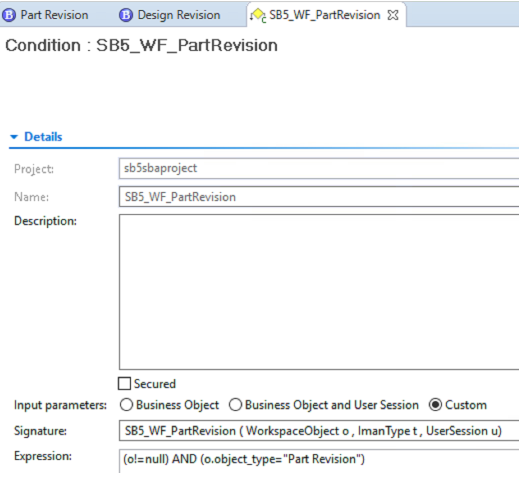
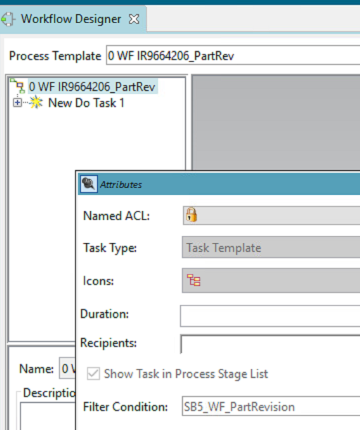
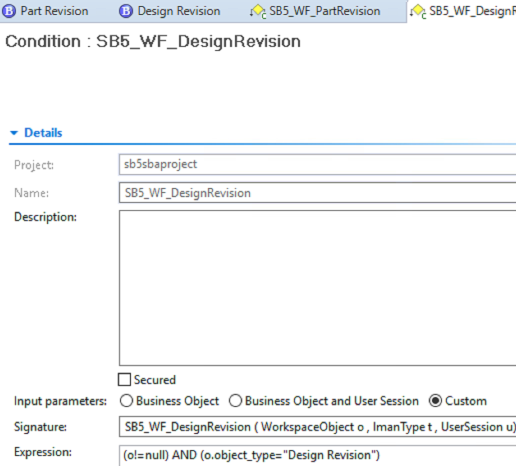
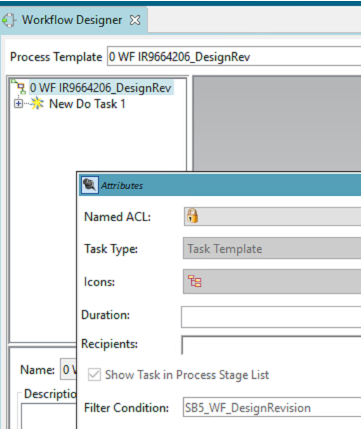
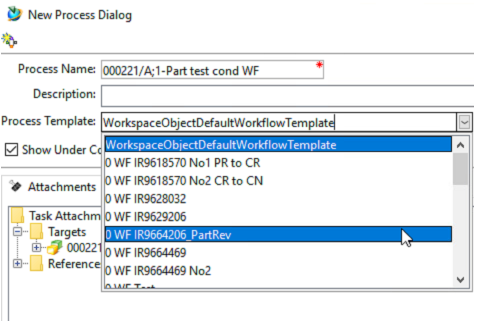
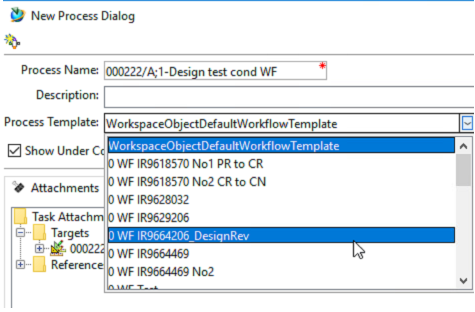
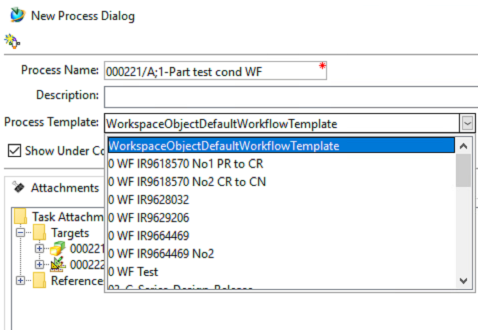
}

# Process template filter / Conditions when launching New Process with several Types

## DETAILS

If the user selects an object in the Teamcenter Rich Client and then calls the "Create New Process" dialog via CTRL+P or from the main menu to send the selected object into a workflow, then only those workflows that are allowed are offered to him as a select list.  
Which workflow templates can be selected for which object type and for which user group can be configured since recent Teamcente versions in two different ways: via Process Template Filter, which are set in the Workflow Designer and are ultimately reference references, or by BMIDE Conditions.  
  
If a distinct workflow template has been assigned for two different object types (type 1 and type2) and the user selects one object of type 1 and the other of type 2 (multiple selection) before starting the New Process Dialog, then both are displayed in the selection field of the available workflow templates.  
At this point, however, we expect only the workflow templates that have been assigned to the two object types using one of the two paths described above, which are "common" to the two.   
Otherwise, users may start workflows that are actually designed for only one of the two object types.  
  
Steps to reproducibility of the problem:  
  
1. log in as dba user via rich client, e.g. as infodba  
  
2. Set the reference WRKFLW\_use\_legacy\_template\_filter to true  
  
3. Set the reference CR\_allow\_alternate\_procedures to Assigned  
  
4. In the Workflow Designer main menu under Edit > Click Template Filter and assign the workflow template "TCM Release Process" for the "Engineering" group for the object type "Design Revision", assign the workflow template "WorkspaceObjectDefaultWorkflowTemplate" for the object type "Part Revision"  
  
5. Sign out and re-log on as a user in the engineering group using rich client  
  
6. Create a Part object and an object of type Design  
  
7. Select the revisions of the items created in the last step by pressing the CTRL key (multiple) and start the "Create New Process" dialog by key combination CTRL + P  
  
>>> Both OOTB workflows configured in the third step are now displayed in the selection list of workflow templates  
  
Expected result:  
No workflows are displayed because one has been configured for themes only and the other is configured only for parts. If a third workflow had been configured for both types, only that one workflow would have been displayed in multiple selection.  
Can this behavior be enforced and, if so, how?

## SOLUTION

1- With legacy "Process template filter" solution:  
  
Indeed in case you select several Item types, the filtered workflow will be displayed in the list even if one of the Item types is not defined for the filter .  
This behavior is as designed.  
  
But these workflow filters are just used as display filters, not to block the submission.  
In your use case you would be able to block the submission of the wrong Item by adding the "EPM-validate-target-objects" rule handler.  
EPM-validate-target-objects  
Description:  
Restricts the types of objects that can be added as target objects.  
  
By adding this rule handler on the Start action in the root of the workflow and with the "-exclude\_type" argument, you will be able to block the submission.  
  
2- With new solution based on BMide conditions:  
  
-BMide condition for the type "Part Revision":  
  
  
  
-BMide condition for type "Design Revision":  
  
  
  
  
  
-If Type PartRev is selected as target, the workflow associated with Type DesignRev is not visible:  
  
  
  
-If Type DesignRev is selected as target, the workflow associated with type PartRev is not visible:  
  
  
  
-If both types are selected as target, no workflow (with associated BMide condition) is visible:  
  


#  Sample ITK function : create saved variant rule

#include <tccore/grm.h>

#include <tccore/grmtype.h>

#include <bom/bom.h>

#include <ps/vrule.h>

static void create\_saved\_variant\_rule(tag\_t window, tag\_t rev\_tag, char \*target\_value)

{

int ifail = ITK\_ok;

tag\_t vrule\_tag = NULLTAG;

ifail = BOM\_window\_ask\_variant\_rule(window, &vrule\_tag);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

int n\_options = 0;

tag\_t \*options = NULL;

tag\_t \*option\_revs = NULL;

ifail = BOM\_variant\_rule\_**ask\_options**(vrule\_tag, &n\_options, &options , &option\_revs);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

/\* assuming only one \*/

tag\_t opt\_tag = options[0];

tag\_t opt\_rev\_tag = option\_revs[0];

if(options) MEM\_free(options);

if(option\_revs) MEM\_free(option\_revs);

int n\_values = 0;

int \*index = NULL;

ifail = BOM\_list\_option\_rev\_values(opt\_rev\_tag, &n\_values, &index);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

for(int ii = 0; ii < n\_values; ii++)

{

char \*value = NULL;

ifail = BOM\_**ask\_option**\_rev\_value(opt\_rev\_tag, index[ii], &value);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

if(strcmp(target\_value, value) == 0)

{

ifail = BOM\_variant\_rule\_set\_option\_values(vrule\_tag , opt\_tag, 1, &index[ii]);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

tag\_t saved\_vrule\_tag = NULLTAG;

ifail = VRULE\_create\_from\_variant\_rule("MyVarRule", "", vrule\_tag, 1, &opt\_tag, &saved\_vrule\_tag);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

tag\_t relation\_type\_tag = NULLTAG;

ifail = GRM\_find\_relation\_type( "IMAN\_specification", &relation\_type\_tag);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

tag\_t relation\_tag = NULLTAG;

ifail = GRM\_create\_relation(rev\_tag, saved\_vrule\_tag , relation\_type\_tag, NULLTAG, &relation\_tag);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

ifail = GRM\_save\_relation(relation\_tag);

if (ifail != ITK\_ok) { /\* your error logic here \*/ }

}

if(value) MEM\_free(value);

}

if(index) MEM\_free(index);

}

# Sample ITK function : create modular variant persistent sos

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

There are two kinds of option sets:

- Runtime or selected option set is a instance of type BOMsos.

Every BOMLine has a selected option set.

- Persistent or stored option set is a database instance of type

StoredOptionSet.

This sample creates a stored option set.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include <bom/bom.h>

#include <tccore/aom\_prop.h>

#include <tccore/grm.h>

static void create\_modular\_variant\_persistent\_sos(tag\_t window,

char \*opt\_name, char \*value)

{

tag\_t top\_line = NULLTAG;

IFERR\_ABORT(BOM\_ask\_window\_top\_line(window, &top\_line));

tag\_t rt\_sos = NULLTAG;

IFERR\_ABORT(BOM\_line\_ask\_sos(top\_line, &rt\_sos));

tag\_t var\_config = NULLTAG;

IFERR\_ABORT(BOM\_create\_variant\_config(NULLTAG, 1, &rt\_sos, &var\_config));

tag\_t item = NULLTAG;

IFERR\_ABORT(AOM\_ask\_value\_tag(top\_line, "bl\_item", &item));

int opt = 0;

IFERR\_ABORT(BOM\_item\_**ask\_option**\_handle(window, item, opt\_name, &opt));

int how\_set = BOM\_option\_set\_by\_user;

IFERR\_ABORT(BOM\_sos\_set\_entry\_string(rt\_sos, opt, "", value, how\_set));

int opt\_name\_length = strlen(opt\_name) + strlen(value) + 2;

char \*db\_sos\_name = NULL;

db\_sos\_name = (char \*) MEM\_alloc(opt\_name\_length \* sizeof(char));

sprintf(db\_sos\_name, "%s - %s", opt\_name, value);

tag\_t db\_sos = NULLTAG;

IFERR\_ABORT(BOM\_sos\_db\_create(db\_sos\_name, var\_config, &db\_sos));

IFERR\_ABORT(AOM\_save(db\_sos));

MEM\_free(db\_sos\_name);

/\* To save sos as References (IMAN\_refrence) to the revision \*/

tag\_t revision = NULLTAG;

IFERR\_ABORT(AOM\_ask\_value\_tag(top\_line, "bl\_revision", &revision));

tag\_t rel\_type = NULLTAG;

IFERR\_ABORT(GRM\_find\_relation\_type( "IMAN\_reference", &rel\_type));

tag\_t relation = NULLTAG;

IFERR\_ABORT(GRM\_create\_relation(revision, db\_sos, rel\_type, NULLTAG, &relation));

IFERR\_ABORT(GRM\_save\_relation(relation));

}

# Creation and management of product configurator objects through **ITK** customization

 Hardware/Software Configuration

 Platform: INTL64

OS: SUSE

OS Version: SLED11S3

Product: TEAMCENTER

Application: CONFIGURATOR

Version: V10.1.7.1

Function: API

 Solution

 Administrator can use standard API for **creating** context and other configurator objects.  
   
**For Product context use ITK**: "ITEM\_create\_item"  
Example:  
ITEM\_create\_item ( "", "ModelFamilyWSO", "Cfg0ProductItem", "A", item1, item\_rev1 )  
   
**For Option Families, values etc**. use TCTYPE\_create\_object **ITK**.  
Example:  
TCTYPE\_ask\_type ( "Cfg0ProductModelFamily", wsoType )  
TCTYPE\_construct\_create\_input( wsoType, createInputTagWso )  
   
**@\* populate with values**  
AOM\_set\_value\_string( createInputTagWso, "cfg0ObjectId", "Engines")  
AOM\_set\_value\_string( createInputTagWso, "object\_name", "Engine Models")  
AOM\_set\_value\_logical( createInputTagWso, "cfg0IsDiscretionary", false)  
AOM\_set\_value\_logical( createInputTagWso , "cfg0HasFreeFormValues", false)  
AOM\_set\_value\_logical( createInputTagWso , "cfg0IsMultiselect", false)  
AOM\_set\_value\_tag( createInputTagWso, "cfg0OwningProductItem", item1 )  
AOM\_set\_value\_string( createInputTagWso, "cfg0FamilyNamespace", itemId )  
AOM\_set\_value\_string( createInputTagWso , "cfg0ValueDataType", CFG0CONFIGURATOR\_String\_Value\_Type )  
   
**@\* create object and save**  
TCTYPE\_create\_object( createInputTagWso, boTag )  
   
   
**To link an option family/value or group to a context you need to allocate them to context.**  
   
Example:  
@\* **Create** allocation record for family  
TCTYPE\_ask\_type ( "Cfg0Allocation", allocationType )  
TCTYPE\_construct\_create\_input( allocationType, allocationCreInput )  
AOM\_set\_value\_string( allocationCreInput, "cfg0ObjectId", "FamilyAllocation" )  
AOM\_set\_value\_string( allocationCreInput, "object\_name", "Family Allocation" )  
AOM\_set\_value\_tag( allocationCreInput, "cfg0ProductItem", itemWSO )  
AOM\_set\_value\_tag( allocationCreInput, "cfg0Target", optionFamilyWSOTag )  
TCTYPE\_create\_object( allocationCreInput, familyAllocationTag )  
AOM\_save( familyAllocationTag )  
AOM\_refresh( familyAllocationTag, 0 )  
   
 **Configurator Rules:**  
   
@\* **Create** Product Item  
TCTYPE\_ask\_type ( "Item", itemType )  
TCTYPE\_construct\_create\_input( itemType, itemCreateInputTag )  
   
AOM\_set\_value\_string( itemCreateInputTag, "object\_name", "product\_item" )  
TCTYPE\_create\_object( itemCreateInputTag, productItemTag )  
AOM\_save\_with\_extensions( productItemTag )  
   
   
@\* **Create**  Default Rule Object without providing required fields.  
TCTYPE\_ask\_type ( "Cfg0DefaultRule", defaultRuleType )  
TCTYPE\_construct\_create\_input( defaultRuleType, defaultRuleCreateInputTag )  
   
@\* **Create**  Default Rule Object by providing required attributes  
@\* and without providing any Thread input descriptor.  
AOM\_set\_value\_string( defaultRuleCreateInputTag, "object\_name", "Default-Rule")  
AOM\_set\_value\_tags( defaultRuleCreateInputTag, "cfg0ProductItems", 1 , { productItemTag } )  
   
@\* **Create** rule object and save. This **creates** thread object in back ground.  
TCTYPE\_create\_object( defaultRuleCreateInputTag, defaultRuleTag )  
AOM\_save( defaultRuleTag )  
AOM\_refresh( defaultRuleTag, 0 )

 Reference

 IR 9082663

#  Sample Teamcenter C++ function : new root item from template using saved **variant** **rule**

#include <iostream>

#include <bom/bom.h>

#include <cfm/cfm.h>

#include <collabctx/collabctx.h>

#include <me/me.h>

#include <tc/tc.h>

#include <tc/tc\_startup.h>

#include <tccore/item.h>

#include <tccore/grm.h>

#include <tccore/grmtype.h>

#include <tccore/tctype.h>

#include <user\_exits/user\_exits.h>

#include <base\_utils/TcResultStatus.hxx>

#include <base\_utils/IFail.hxx>

using namespace std;

static int new\_root\_item\_from\_template\_using\_saved\_**variant\_rule**(tag\_t tRev, tag\_t tSVR)

{

int ifail = **ITK**\_ok;

ResultStatus status;

try

{

tag\_t tWindow = NULLTAG;

status = BOM\_create\_window (&tWindow);

tag\_t tTopLine = NULLTAG;

status = BOM\_set\_window\_top\_line(tWindow, NULLTAG, tRev, NULLTAG, &tTopLine);

tag\_t t**Rule** = NULLTAG;

status = CFM\_find("Latest Working", &t**Rule**);

status = BOM\_set\_window\_config\_**rule**(tWindow, t**Rule**);

status = BOM\_window\_hide\_**variants** (tWindow);

status = BOM\_window\_apply\_full\_vrule(tWindow, tSVR);

tag\_t ccContext = NULLTAG;

status = COLLABCTX\_create\_config\_context\_based\_on\_ref( "", "CCName", "CCDesc", &tWindow, &ccContext);

tag\_t tNewWindow = NULLTAG;

status = BOM\_create\_window (&tNewWindow);

tag\_t tItem = NULLTAG;

status = ITEM\_ask\_item\_of\_rev(tRev, &tItem);

tag\_t tType =NULLTAG;

status = TCTYPE\_ask\_object\_type(tItem, &tType);

logical isModified = FALSE;

char\* itemId = NULL;

status = USER\_new\_item\_id(NULLTAG, tType, &isModified, &itemId);

cout << "\n itemId: "<< itemId << endl;

tag\_t tNewRev;

char **rule**Key[] = "Product.Template.Cloning";

status = ME\_create\_product\_from\_template(itemId, "A", itemId, "", tRev, ccContext, tNewWindow, **rule**Key, &tNewRev);

status = BOM\_close\_window(tWindow);

status = BOM\_close\_window(tNewWindow);

if(itemId) MEM\_free(itemId);

}

catch( const IFail &e )

{

ifail = e.ifail();

cout << "\n error " << ifail << endl;

cout << " "<< e.getMessage() << endl;

}

return ifail;

}

#  Sample Teamcenter C++ function : new child item from template using saved **variant** **rule**

#include <iostream>

#include <bom/bom.h>

#include <cfm/cfm.h>

#include <collabctx/collabctx.h>

#include <me/me.h>

#include <tc/tc.h>

#include <tc/tc\_startup.h>

#include <tccore/item.h>

#include <tccore/grm.h>

#include <tccore/grmtype.h>

#include <tccore/tctype.h>

#include <user\_exits/user\_exits.h>

#include <base\_utils/TcResultStatus.hxx>

#include <base\_utils/IFail.hxx>

using namespace std;

static int new\_child\_item\_from\_template\_using\_saved\_**variant\_rule**(tag\_t tRev, tag\_t tSVR)

{

int ifail = **ITK**\_ok;

ResultStatus status;

try

{

tag\_t tWindow = NULLTAG;

status = BOM\_create\_window (&tWindow);

tag\_t tTopLine = NULLTAG;

status = BOM\_set\_window\_top\_line(tWindow, NULLTAG, tRev, NULLTAG, &tTopLine);

tag\_t t**Rule** = NULLTAG;

status = CFM\_find("Latest Working", &t**Rule**);

status = BOM\_set\_window\_config\_**rule**(tWindow, t**Rule**);

status = BOM\_window\_hide\_**variants** (tWindow);

status = BOM\_window\_apply\_full\_vrule(tWindow, tSVR);

tag\_t ccContext = NULLTAG;

status = COLLABCTX\_create\_config\_context\_based\_on\_ref( "", "CCName", "CCDesc", &tWindow, &ccContext);

tag\_t tItem = NULLTAG;

status = ITEM\_ask\_item\_of\_rev(tRev, &tItem);

tag\_t tType =NULLTAG;

status = TCTYPE\_ask\_object\_type(tItem, &tType);

logical isModified = FALSE;

char\* itemId = NULL;

status = USER\_new\_item\_id(NULLTAG, tType, &isModified, &itemId);

cout << "\n itemId: "<< itemId << endl;

tag\_t tNewRev;

char **rule**Key[] = "Product.Template.Cloning";

status = ME\_create\_product\_from\_template(itemId, "A", itemId, "", tRev, ccContext, tWindow, **rule**Key, &tNewRev);

status = BOM\_close\_window(tWindow);

if(itemId) MEM\_free(itemId);

}

catch( const IFail &e )

{

ifail = e.ifail();

cout << "\n error " << ifail << endl;

cout << " "<< e.getMessage() << endl;

}

return ifail;

}

# Sample **ITK** function : list **secondary** objects

#include <**itk**/mem.h>

#include <tccore/aom\_prop.h>

#include <tccore/grm.h>

#include <tccore/tctype.h>

void list\_displayable\_properties\_with\_value(char \*indention, tag\_t **object**)

{

logical

is\_displayable = TRUE;

int

n\_props = 0,

ii = 0;

char

\*\*prop\_names = NULL,

\*disp\_name = NULL,

\*value = NULL;

IFERR\_REPORT( AOM\_ask\_prop\_names(**object**, &n\_props, &prop\_names) );

for( ii = 0; ii < n\_props; ii++)

{

IFERR\_REPORT( AOM\_UIF\_is\_displayable(**object**, prop\_names[ii],

&is\_displayable));

if (is\_displayable == TRUE)

{

value = NULL;

IFERR\_REPORT( AOM\_UIF\_ask\_name(**object**, prop\_names[ii], &disp\_name) );

IFERR\_REPORT( AOM\_UIF\_ask\_value(**object**, prop\_names[ii], &value) );

if ( (value != NULL) && (strlen(value) > 0 ) )

{

if (strlen(value) == 1 )

{

if ( strcmp(value, " ") != 0 )

fprintf(stdout, "%s %s: %s \n", indention, disp\_name,

value );

}

else

fprintf(stdout, "%s %s: %s\n", indention, disp\_name, value);

}

}

}

if (prop\_names != NULL) MEM\_free(prop\_names);

if (disp\_name != NULL) MEM\_free(disp\_name);

if (value != NULL) MEM\_free(value);

}

void list\_secondary\_objects(tag\_t primary\_object, char \*rel\_type\_name)

{

ECHO("\n\n list\_secondary\_objects \n\n");

tag\_t relation\_type = NULLTAG;

if ((rel\_type\_name != NULL) && (strlen(rel\_type\_name) > 0 ))

{

IFERR\_REPORT(GRM\_find\_relation\_type(rel\_type\_name, &relation\_type));

}

int n\_objects = 0;

tag\_t \***objects** = NULL;

IFERR\_REPORT(GRM\_list\_secondary\_objects\_only(primary\_object, relation\_type,

&n\_objects, &**objects**));

ECHO(" n\_objects: %d \n", n\_objects);

for(int ii = 0; ii < n\_objects; ii++)

{

list\_displayable\_properties\_with\_value(" ", **objects**[ii]);

}

ECHO(" \n");

MEM\_free(**objects**);

}

# Sample **ITK** function : va list for GRM copy

int ImanRelationCopy(METHOD\_message\_t \*msg,va\_list args)

{

int ifail = **ITK**\_ok;

va\_list largs;

va\_copy( largs, args );

/\* va\_list for GRM\_copy\_msg \*/

tag\_t primary\_**object** = va\_arg(largs, tag\_t);

tag\_t **secondary\_object** = va\_arg(largs, tag\_t);

tag\_t relation\_type = va\_arg(largs, tag\_t);

tag\_t user\_data = va\_arg(largs, tag\_t);

tag\_t \*new\_relation = va\_arg(largs, tag\_t \*);

va\_end(largs);

is\_instance\_in\_database("primary\_**object**", primary\_**object**);

is\_instance\_in\_database("**secondary\_object**", **secondary\_object**);

is\_instance\_in\_database("\*new\_relation", \*new\_relation);

return ifail;

}

# Sample **ITK** function : demo removal of certain reference types

#include <epm/epm.h>

#include <pom/pom/pom.h>

#include <tc/folder.h>

#include <tccore/aom.h>

#include <tccore/grm.h>

#include <tccore/grmtype.h>

#include <tccore/tctype.h>

#include <tccore/workspaceobject.h>

static logical is\_descendant\_of\_folder(tag\_t object\_tag)

{

tag\_t parent\_class = NULLTAG;

IFERR\_REPORT(POM\_class\_id\_of\_class("Folder", &parent\_class));

tag\_t class\_tag = NULLTAG;

IFERR\_REPORT(POM\_class\_of\_instance(object\_tag, &class\_tag));

logical verdict = FALSE;

IFERR\_REPORT(POM\_is\_descendant(parent\_class, class\_tag, &verdict));

return verdict;

}

static void demo\_removal\_of\_certain\_reference\_types(tag\_t object\_tag)

{

int n\_references = 0;

int \*levels = NULL;

tag\_t \*reference\_tags = NULL;

char \*\*relation\_type\_name = NULL;

IFERR\_REPORT(WSOM\_where\_referenced(object\_tag, 1, &n\_references, &levels,

&reference\_tags, &relation\_type\_name));

for (int ii = 0; ii < n\_references; ii++)

{

char type\_name[WSO\_name\_size\_c + 1] = "";

IFERR\_REPORT(WSOM\_ask\_object\_type(reference\_tags[ii], type\_name));

if (is\_descendant\_of\_folder(reference\_tags[ii]))

{

tag\_t folder = reference\_tags[ii];

IFERR\_REPORT(AOM\_refresh(folder, TRUE));

IFERR\_REPORT(FL\_remove(folder, object\_tag));

IFERR\_REPORT(AOM\_save(folder));

IFERR\_REPORT(AOM\_refresh(folder, FALSE));

}

else if ( (relation\_type\_name[ii] != NULL) &&

(strlen(relation\_type\_name[ii]) > 0 ) )

{

tag\_t relation\_type\_tag = NULLTAG;

IFERR\_REPORT(GRM\_find\_relation\_type(relation\_type\_name[ii],

&relation\_type\_tag));

// GRM\_delete\_relation requires the relation **object** which we don't have

tag\_t relation\_tag = NULLTAG;

// First try relation using the reference as a primary **object**

tag\_t primary\_object = reference\_tags[ii];

tag\_t second\_object = object\_tag;

IFERR\_REPORT(GRM\_find\_relation(primary\_object, second\_object,

relation\_type\_tag, &relation\_tag));

// If no relation is found try the reference as a **secondary** **object**

if (relation\_tag == NULLTAG)

{

primary\_object = object\_tag;

second\_object = reference\_tags[ii];

IFERR\_REPORT(GRM\_find\_relation(primary\_object, second\_object,

relation\_type\_tag, &relation\_tag));

}

IFERR\_REPORT(GRM\_delete\_relation(relation\_tag ));

}

else if(strcmp(type\_name, "EPMTask") == 0 )

{

tag\_t task = reference\_tags[ii];

EPM\_state\_t state;

IFERR\_REPORT(EPM\_ask\_state(task, &state));

char state\_string[WSO\_name\_size\_c + 1] = "";

IFERR\_REPORT(EPM\_ask\_state\_string(state, state\_string));

if(strcmp(state\_string, "Completed") == 0)

{

ECHO("\n Can't remove targets of completed jobs! \n");

}

else

{

ITK\_set\_bypass(TRUE);

IFERR\_REPORT(EPM\_remove\_attachments(task, 1, &object\_tag));

}

}

}

if(levels) MEM\_free(levels);

if(reference\_tags) MEM\_free(reference\_tags);

if(relation\_type\_name) MEM\_free(relation\_type\_name);

}

# Sample **ITK** function : relation create pre action

/\*HEAD RELATION\_CREATE\_PRE\_ACTION CCC **ITK** \*/

extern DLLAPI int relation\_create\_pre\_action(METHOD\_message\_t \*msg, va\_list args)

{

ECHO( "\n relation\_create\_pre\_condition \n");

ECHO( "\n ImanRelation - Create - GRM\_create\_msg \n");

int ifail = ITK\_ok;

va\_list largs;

va\_copy( largs, args );

tag\_t primary\_object = va\_arg(largs, tag\_t);

tag\_t secondary\_object = va\_arg(largs, tag\_t);

tag\_t relation\_type = va\_arg(largs, tag\_t);

tag\_t user\_data = va\_arg(largs, tag\_t);

tag\_t \*new\_relation = va\_arg(largs, tag\_t \*);

va\_end( largs );

ECHO(" primary\_object: %u\n", primary\_object);

ECHO(" secondary\_object: %u\n", secondary\_object);

ECHO(" relation\_type: %u\n", relation\_type);

ECHO(" user\_data: %u\n", user\_data);

ECHO(" new\_relation: %u\n", \*new\_relation);

return ifail;

}

# Creation and management of product configurator objects through Heerstraße, 14052 Berlin**ITK** customization

 We want to know if there are any **ITKs** available to **create** product configurator objects such as Configurator context, Family group, **Option** family, **options** and saved variant rule from outside of Teamcenter

 Hardware/Software Configuration

 Platform: INTL64

OS: SUSE

OS Version: SLED11S3

Product: TEAMCENTER

Application: CONFIGURATOR

Version: V10.1.7.1

Function: API

 Solution

 Administrator can use standard API for **creating** context and other configurator objects.  
   
**For Product context use ITK**: "ITEM\_create\_item"  
Example:  
ITEM\_create\_item ( "", "ModelFamilyWSO", "Cfg0ProductItem", "A", item1, item\_rev1 )  
   
**For Option Families, values etc**. use TCTYPE\_create\_object **ITK**.  
Example:  
TCTYPE\_ask\_type ( "Cfg0ProductModelFamily", wsoType )  
TCTYPE\_construct\_create\_input( wsoType, createInputTagWso )  
   
**@\* populate with values**  
AOM\_set\_value\_string( createInputTagWso, "cfg0ObjectId", "Engines")  
AOM\_set\_value\_string( createInputTagWso, "object\_name", "Engine Models")  
AOM\_set\_value\_logical( createInputTagWso, "cfg0IsDiscretionary", false)  
AOM\_set\_value\_logical( createInputTagWso , "cfg0HasFreeFormValues", false)  
AOM\_set\_value\_logical( createInputTagWso , "cfg0IsMultiselect", false)  
AOM\_set\_value\_tag( createInputTagWso, "cfg0OwningProductItem", item1 )  
AOM\_set\_value\_string( createInputTagWso, "cfg0FamilyNamespace", itemId )  
AOM\_set\_value\_string( createInputTagWso , "cfg0ValueDataType", CFG0CONFIGURATOR\_String\_Value\_Type )  
   
**@\* create object and save**  
TCTYPE\_create\_object( createInputTagWso, boTag )  
   
   
**To link an option family/value or group to a context you need to allocate them to context.**  
   
Example:  
@\* **Create** allocation record for family  
TCTYPE\_ask\_type ( "Cfg0Allocation", allocationType )  
TCTYPE\_construct\_create\_input( allocationType, allocationCreInput )  
AOM\_set\_value\_string( allocationCreInput, "cfg0ObjectId", "FamilyAllocation" )  
AOM\_set\_value\_string( allocationCreInput, "object\_name", "Family Allocation" )  
AOM\_set\_value\_tag( allocationCreInput, "cfg0ProductItem", itemWSO )  
AOM\_set\_value\_tag( allocationCreInput, "cfg0Target", optionFamilyWSOTag )  
TCTYPE\_create\_object( allocationCreInput, familyAllocationTag )  
AOM\_save( familyAllocationTag )  
AOM\_refresh( familyAllocationTag, 0 )  
   
 **Configurator Rules:**  
   
@\* **Create** Product Item  
TCTYPE\_ask\_type ( "Item", itemType )  
TCTYPE\_construct\_create\_input( itemType, itemCreateInputTag )  
   
AOM\_set\_value\_string( itemCreateInputTag, "object\_name", "product\_item" )  
TCTYPE\_create\_object( itemCreateInputTag, productItemTag )  
AOM\_save\_with\_extensions( productItemTag )  
   
   
@\* **Create**  Default Rule Object without providing required fields.  
TCTYPE\_ask\_type ( "Cfg0DefaultRule", defaultRuleType )  
TCTYPE\_construct\_create\_input( defaultRuleType, defaultRuleCreateInputTag )  
   
@\* **Create**  Default Rule Object by providing required attributes  
@\* and without providing any Thread input descriptor.  
AOM\_set\_value\_string( defaultRuleCreateInputTag, "object\_name", "Default-Rule")  
AOM\_set\_value\_tags( defaultRuleCreateInputTag, "cfg0ProductItems", 1 , { productItemTag } )  
   
@\* **Create** rule object and save. This **creates** thread object in back ground.  
TCTYPE\_create\_object( defaultRuleCreateInputTag, defaultRuleTag )  
AOM\_save( defaultRuleTag )  
AOM\_refresh( defaultRuleTag, 0 )

 Reference

 IR 9082663

#  Sample **ITK** function : report referencers of workspaceobject

#include <**itk**/mem.h>

#include <tccore/workspaceobject.h>

static void report\_referencers\_of\_workspaceobject(tag\_t wso)

{

int n\_references = 0, \*levels = NULL;

tag\_t \*reference\_tags = NULL;

char \*\***relations** = NULL;

IFERR\_REPORT(WSOM\_where\_referenced(wso, 1 , &n\_references, &levels,

&reference\_tags,&**relations**));

printf("\n n\_references: %d \n", n\_references);

for (int ii = 0; ii < n\_references; ii++)

{

char \*id = NULL;

IFERR\_REPORT(WSOM\_ask\_object\_id\_string(reference\_tags[ii], &id));

char type[WSO\_name\_size\_c + 1] = "";

IFERR\_REPORT(WSOM\_ask\_object\_type(reference\_tags[ii], type));

if ( (**relations**[ii] != NULL) && (strlen(**relations**[ii]) > 0 ) )

{

printf(" %s (%s) - **Relation**: %s \n", id, type, **relations**[ii]);

}

else printf(" %s (%s) \n", id, type);

MEM\_free(id);

}

MEM\_free(levels);

MEM\_free(reference\_tags);

MEM\_free(**relations**);

}

#  Sample **ITK** function : find item revisions completed jobs

#include <epm/epm.h>

#include <**itk**/mem.h>

#include <pom/pom/pom.h>

#include <tccore/workspaceobject.h>

static void find\_item\_revisions\_completed\_jobs(tag\_t item\_revision)

{

int n\_references = 0, \*levels = NULL;

tag\_t \*reference\_tags = NULL;

char \*\***relations** = NULL;

IFERR\_REPORT(WSOM\_where\_referenced(item\_revision, 1, &n\_references, &levels,

&reference\_tags,&**relations**));

printf("\n n\_references: %d \n", n\_references);

for (int ii = 0; ii < n\_references; ii++)

{

tag\_t class\_id = NULLTAG;

IFERR\_REPORT(POM\_class\_of\_instance(reference\_tags[ii], &class\_id));

char\* class\_name = NULL;

IFERR\_REPORT(POM\_name\_of\_class(class\_id, &class\_name));

if (!strcmp(class\_name,"EPMTask"))

{

char task\_name[WSO\_name\_size\_c+1] = "";

IFERR\_REPORT(EPM\_ask\_name(reference\_tags[ii], task\_name));

EPM\_state\_t state;

IFERR\_REPORT(EPM\_ask\_state(reference\_tags[ii], &state));

char state\_string[WSO\_name\_size\_c+1] = "";

IFERR\_REPORT(EPM\_ask\_state\_string(state, state\_string));

printf("%s - %s\n", task\_name, state\_string);

}

if(class\_name) MEM\_free(class\_name);

}

if(levels) MEM\_free(levels);

if(reference\_tags) (reference\_tags);

if(**relations**) MEM\_free(**relations**);

}

Sample ITK function : create relation with required property

#include <tccore/aom.h>

#include <tccore/aom\_prop.h>

#include <tccore/tctype.h>

static void create\_relation\_with\_required\_property(tag\_t primary\_object,

tag\_t secondary\_object, tag\_t relation\_type)

{

tag\_t grm\_type = NULLTAG;

IFERR\_REPORT(TCTYPE\_find\_type("A2ImanRelation", NULL, &grm\_type));

tag\_t grm\_create\_input = NULLTAG;

IFERR\_REPORT(TCTYPE\_construct\_create\_input(grm\_type, &grm\_create\_input));

IFERR\_REPORT(AOM\_set\_value\_tag(grm\_create\_input, "primary\_object",

primary\_object));

IFERR\_REPORT(AOM\_set\_value\_tag(grm\_create\_input, "secondary\_object",

secondary\_object));

IFERR\_REPORT(AOM\_set\_value\_tag(grm\_create\_input, "relation\_type",

relation\_type));

IFERR\_REPORT(AOM\_set\_value\_string(grm\_create\_input, "a2\_required\_prop",

"some string"));

tag\_t **relation** = NULLTAG;

IFERR\_REPORT(TCTYPE\_create\_object(grm\_create\_input, &**relation**));

IFERR\_REPORT(AOM\_save(**relation**));

}