ORACLE
CPQ Cloud

Contract Negotiation

Custom Solution Setup

ORACLE TECHNICAL WHITE PAPER | CPQ 2016 R1

ORACLE

Disclaimer

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

Table of Contents

Disclaimer 1
Introduction 1
Purpose 1
Overview 1
- Compare 1
- Merge 1

Contract Negotiation Sample Flow 1
Build In Commerce 2
- Associated Actions and Attributes 2
  - Commerce Quote Level Actions 2
  - Commerce Quote Level Attributes 3
  - Commerce Library Scripts 3
  - BML Library Scripts 3
  - Email Designer Templates 4
  - Document Designer Templates 4
- Actions Setup 4
  - Modify Actions 4
  - Generate BML 4
  - Check For Amendments BML 9
  - Email Actions 10
- Attributes Setup 12
  - Process JSON to Table BML 12
- BML Library Scripts Setup 14
  - BML Library Scripts 14
  - Get File Attachment Data BML 15
  - Update File Attachment BML 16
  - Get File Id From Response BML 16
  - Print Email Designer BML 16
  - Print Doc Designer BML 18
  - Compare DOCX BML 19
  - Merge DOCX BML 19
  - Get Basic Auth Credentials BML 20
- Data Table Setup 20
  - Credentials 20
- Commerce Layout Setup 21
- Template Setup 21
- Ready To Go 21

Conclusion 21

Introduction

This is an example of a customizable Contract Negotiation setup.

Purpose

To setup the Contract Negotiation Process using tools already existing and newly added in R1 2016. The actions and attributes can then be tailored to your business needs using this configuration as a template.

Overview

Contract Negotiation is made possible by two main components: Compare and Merge. Using these components, it is possible to make a customizable Contract Negotiation solution that will fit your business needs.

Compare

Compare is a REST endpoint that accepts two Commerce File Attachment Attribute IDs as input, and returns with a list of changed areas made between the two documents.

- The files must be DOCX format.
- They must be generated by the same Document Designer Template.
- The Template must be a "Contract" Template.
- As of 2016 R1, response will only include changes to the Highest Level of object, so the list of changes is not exhaustive, but is *maximally inclusive*.

Merge

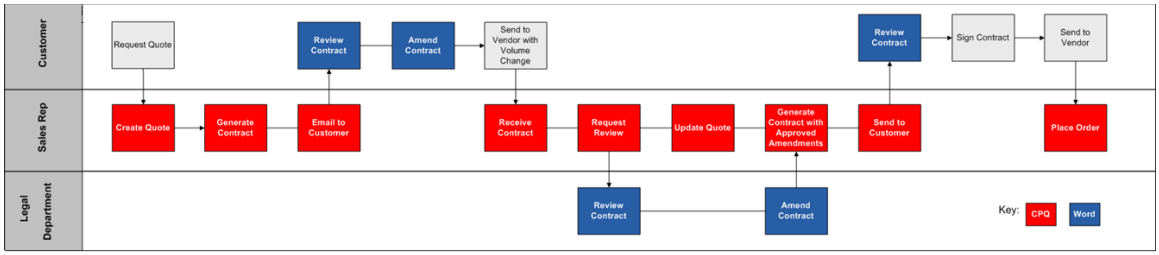
Merge is a REST endpoint that accepts a source document Commerce File Attachment ID and a merge target document Commerce File Attachment ID, as well as a list of Approved Changes as identified from a Compare operation. Merge will respond with the updated document represented by a Base64 encoded string.

- The files must be DOCX format.
- They must be generated by the same Document Designer Template.
- The Template must be a "Contract" Template.
- As of 2016 R1, response will only include changes to the Highest Level of object, so the list of changes is not exhaustive, but is *maximally inclusive*.

Contract Negotiation Sample Flow

After a Sales User has determined it is time to generate a Contract for their Customer they can click on the "Generate" action on the Transaction page. This will populate the "Customer Contract" File Attachment. They can now email this Contract to their Customer using the "Send to Customer" action. The Customer may want to make some modifications to the Contract, which they will perform in Microsoft Word. After modifying, the Customer will send the Contract back to the Sales User, and they will upload it the "Customer Contract" File Attachment. The "Check For Amendments" action can now be used to generate a table with High Level changes made by the Customer, such as which clause was modified, and whether any were removed or inserted. The table will also

indicate whether the changes were Accepted using Microsoft Word's Track Changes feature. Once the Sales User is satisfied with the Customer changes, the "Customer Contract" File Attachment should be updated with that version. At this point the Sales User will click on the "Generate" action again, this time any changes made in the Document Designer Template, or in the Transaction that may generate different content, will be Merged into the "Customer Contract". Changes that were made by the Customer or the Sales User that conflict with the changes made by "Generate" will be overridden by "Generate". Now the Sales User is ready to send the "Customer Contract" to his Legal team. At this point, the Contract Negotiation cycle is completed, or ready to start again with Legal changes.



Contract Negotiation Cycle

Build In Commerce

Associated Actions and Attributes

Commerce Quote Level Actions

Label	Variable Name	Type	Description
Check For Amendments	ORCL_CN_CheckForAmendments	Modify	Creates an HTML Table with a list of changes between the Customer Contact File Attachment and the Generated Contract File Attachment.
Generate	ORCL_CN_Generate	Modify	Generates a "Customer Contract" using the associated Negotiated Contract Template. If the Customer Contract already exists, the File Attachment is updated with the changes between a Previously Generated Contract and a Newly Generated Contract.
Send To Customer	ORCL_CN_SendToCustomer	Email	Sends an Email to the Customer with the "Customer Contract" attached.
Send To Legal	ORCL_CN_SendToLegal	Email	Sends an Email to the Legal Department with the "Customer Contract" attached.
FA Print 1	ORCL_CN_FAPrint1	Print	Print Action Needed For Updating File Attachments Via REST
FA Print 2	ORCL_CN_FAPrint3	Print	Print Action Needed For Updating File Attachments Via REST
FA Print 3	ORCL_CN_FAPrint2	Print	Print Action Needed For Updating File Attachments Via REST
FA Print 4	ORCL_CN_FAPrint4	Print	Print Action Needed For Updating File Attachments Via REST

Commerce Quote Level Attributes

Label	Variable Name	Type	Description
Customer Contract	ORCL_CN_CustomerContract_t	File Attachment	Customer Contract Document
Generated Contract	ORCL_CN_GeneratedContract_t	File Attachment	Contract Generated by "Generate" action.
Previously Generated Contract	ORCL_CN_PreviouslyGeneratedContract_t	File Attachment	Last saved Generated Contract.
Temporary Attachment	ORCL_CN_TempAttachment_t	File Attachment	Stores documents during "Generate" action until Compare and Merge complete successfully.
Signed Contract	ORCL_CN_SignedContract_t	File Attachment	Signed copy of final Contract.
Amendments Display	ORCL_CN_AmendmentsDisplay_t	HTML	HTML Table to display changes returned by "Check For Amendments" action.
Compare JSON	ORCL_CN_CompareJSON_t	Text Area	Stores the JSON String returned by "Check For Amendments" action.
Action Status	ORCL_CN_ActionStatus_t	Text Area	Contains Updates from "Generate" and "Check For Amendments" actions; success or errors.

Commerce Library Scripts

Label	Variable Name	Description
Process JSON To Table	ORCL_CN_ProcessJsonToTable	Process the JSON returned by "Compare" REST endpoint response into a HTML Table for easy viewing by Sales Users.

BML Library Scripts

Label	Variable Name	Description
Get File Attachment Data	ORCL_CN_GetFileAttachmentData	Call the Get File Attachment REST method and returns the File Attachment as a Base64 encoded String.
Update File Attachment	ORCL_CN_UpdateFileAttachment	Updates a File Attachment using a Base64 encoded String as input. Returns the response from the server.
Get File Id From Response	ORCL_CN_GetFileIdFromResponse	Parses the Update File Attachment response for the File ID.
Print Email Designer	ORCL_CN_PrintEmailDesigner	Prints an Email Designer Template and returns with the Subject and Body Strings.
Print Doc Designer	ORCL_CN_PrintDocDesigner	Prints a Document Designer Template and returns as a Base64 encoded String.
Compare Docx	ORCL_CN_CompareDocx	Compares two DOCX documents in File Attachment Attributes and returns the differences as a JSON String.
Merge Docx	ORCL_CN_MergeDocx	Merges changes found in a Source File Attachment DOCX into a Target File Attachment DOCX using a list of changes in JSON String format. Returns the updated DOCX as a Base64 encoded String.
getBasicAuthCredentials	getBasicAuthCredentials	Gets Authorization Credentials from secure columns of a data table. Requires a properly setup data table.

Email Designer Templates

Name	Description
Send To Customer Template	Template used to generate an Email for sending the Customer Contract to the Customer.
Send To Legal Template	Template used to generate an Email for sending the Customer Contract to the Legal Department.

Document Designer Templates

Label	Type	Description
Negotiated Contract	Single Language, Contract, Track Changes	The Document Designer Template that will be used to create the Contract.

Actions Setup

Modify Actions

The Generate and Compare actions should be Modify type and have BML in the Advanced Modify - Before Formulas function.

Generate BML

```
» *   CONTRACT NEGOTIATION  -  GENERATE
» *    prints a contract template specified by "templateName"
» *    performs a comparison between print output and previous print output
» *    if differences occured, merges changes into Customer Contract
» *
» *    updates all attachments with print output if Customer Contract is empty
» */
»
»
» siteName       = _system_company_name + ".us.oracle.com";
» protocol       = "http";
» siteUrl        = protocol+"://"+siteName;
»
» userAuth       = "Basic " + util.getBasicAuthCredentials("cpqFullAccessUser", "User
  Authentication Error.");
»
» processId      = 36244034;       // update for commerce process
» processVarName = "oraclecpqo";   // update for commerce process
» bsId           = atoi(_system_buyside_id);
» fileType       = "docx";
» templateLang   = -1;             // update for template language
» templateName   = "ORCL_CN Negotiated Contract";
»
» prvPrintSId    = ORCL_CN_PreviouslyGeneratedContract_t;
» curPrintSId    = ORCL_CN_GeneratedContract_t;
» custContSId    = ORCL_CN_CustomerContract_t;
»
» actionHistryV  = "ORCL_CN_ActionStatus_t";
```

```
» previusPrintV  = "ORCL_CN_PreviouslyGeneratedContract_t";
» currentPrintV  = "ORCL_CN_GeneratedContract_t";
» customerContV  = "ORCL_CN_CustomerContract_t";
» temporyPrintV  = "ORCL_CN_TempAttachment_t";
»
» base64Key      = "base64doc";
» errorKey       = "error";
»
» curPrintFileN  = "currentPrint.docx";
» prvPrintFileN  = "previousPrint.docx";
» ccdPrintFileN  = "CustomerContract.docx";
»
» prevPrintId    = -1;
» curPrintId     = -1;
» custContId     = -1;
»
» // history attributes do not work in this context because of tilde, please ignore.
  thanks.
» history = "";//datetostr(getdate())+ "~" +_system_user_login +"~"+ "12345" +"~";
»
» restResp       = dict("string");
»
»
»
» // get NEW PRINT OUTPUT of document
» newPrintBase64 = "";
» restResp = util.ORCL_CN_PrintDocDesigner(siteUrl, userAuth, processVarName, bsId,
  templateName, templateLang, "DOCX");
» error = get(restResp, errorKey);
» if(len(error) > 0){
»     return "1~" + actionHistryV + "~"+ history + "Print Error: " + error;
» }else{
»     newPrintBase64 = get(restResp, base64Key);
» }
»
»
»
»
»
» // if Value in CUSTOMER CONTRACT == null, copy OUTPUT Print to all Files
» custContrResp = util.ORCL_CN_GetFileAttachmentData(siteUrl, userAuth, processVarName,
  bsId, customerContV);
» firstPrint = len(get(custContrResp, base64Key)) < 2;
» custContrResp = dict("string");
»
» if(firstPrint){
»
»     // ***  UPDATE FILE ATTACHMENTS *********
»     // -- if no data exist in Customer Contract, copy the printed output to all three
  file attachments.
»     // -- This starts/re-starts the Contract Negotiation Process for this file
  attachment set
»
»     // copy OUTPUT to Prev Print
»     restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, previusPrintV, fileType, newPrintBase64,  prvPrintFileN, true);
»     if(len(get(restResp, errorKey)) > 0){
»         return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error (New
  Previous Print): " + get(restResp, errorKey);
»     }
»   prevPrintId = util.ORCL_CN_GetFileIdFromResponse(restResp);
```

```
»     // copy OUTPUT to Current Print Contract
»     restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, currentPrintV, fileType, newPrintBase64,  curPrintFileN, true);
»     if(len(get(restResp, errorKey)) > 0){
»         return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error (New
  Current Print): " + get(restResp, errorKey);
»     }
»   curPrintId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»     // copy OUTPUT to Customer Contract
»     restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, customerContV, fileType, newPrintBase64,  ccdPrintFileN, true);
»     if(len(get(restResp, errorKey)) > 0){
»         return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error (New
  Customer Contract): " + get(restResp, errorKey);
»     }
»   custContId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»
»     return "1~" + actionHistryV + "~"+ history + "Generate Successful"+
»          "|" + "1~" + customerContV + "~"+ string(custContId)+
»          "|" + "1~" + previusPrintV + "~"+ string(prevPrintId)+
»          "|" + "1~" + currentPrintV + "~"+ string(curPrintId);
»     // ***  ********************** *********
»
»
»
»
»
»
» // NOT FIRST PRINT
» } else {
»     currentPrintResp = util.ORCL_CN_GetFileAttachmentData(siteUrl, userAuth,
  processVarName, bsId, currentPrintV);
»     currentPrintB64 = get(currentPrintResp, base64Key);
»     currentPrintResp = dict("string");
»
»     // ***  UPDATE FILE ATTACHMENTS *********
»     // -- check if Current and Previous have data & copy new data to Temp
»     oldPrintId = -1;
»     updatedCurPrev = false;
»     if(len(currentPrintB64) < 2 OR len(get(restResp, errorKey)) > 0){
»         // nothing in Current Print or could not retrieve it
»         // copy OUTPUT to Current Print
»         restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, currentPrintV,     fileType, newPrintBase64,  curPrintFileN, true);
»         if(len(get(restResp, errorKey)) > 0){
»             return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error
  (Current Print): " + get(restResp, errorKey);
»         }
»         curPrintId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»
»         if(isnumber(prvPrintSId)){
»             // PREVIOUS PRINT
»             // use previous attachment id
»             oldPrintId = atoi(prvPrintSId);
»                  prevPrintId = atoi(prvPrintSId);
»             history = history + "Current Print Updated, using Previous Print for
  Compare.\n";
»         }else{
»             restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth,
  processVarName, bsId, previusPrintV,     fileType, newPrintBase64,  prvPrintFileN,
  true);
```

```
»             if(len(get(restResp, errorKey)) > 0){
»                 return "1~" + actionHistryV + "~"+ history +"Update File Attachment
  Error (Current Print): " + get(restResp, errorKey);
»             }
»             updatedCurPrev = true;
»                  prevPrintId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»         }
»     }else{
»          curPrintId = atoi(curPrintSId);
»          prevPrintId = atoi(prvPrintSId);
»
»         currentPrintB64 = "";
»         // copy OUTPUT to Temp
»         restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, temporyPrintV,     fileType, newPrintBase64,  "tempPrint.docx", true);
»         if(len(get(restResp, errorKey)) > 0){
»             return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error
  (Temp Print): " + get(restResp, errorKey);
»         }
»     }
»
»     if(updatedCurPrev){
»         return "1~" + actionHistryV + "~"+ history +"Updated Current and Previous
  Print." +
»           "|" + "1~" + customerContV + "~"+ custContSId +
»           "|" + "1~" + previusPrintV + "~"+ string(prevPrintId)+
»           "|" + "1~" + currentPrintV + "~"+ string(curPrintId);
»     }
»     // ***  ********************** *********
»
»
»     if(isnumber(curPrintSId)){ // use current attachement id
»         oldPrintId = atoi(curPrintSId); // Current PRINT
»     }
»     newPrintId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»     custContId = atoi(custContSId); // CUSTOMER CONTRACT
»
»
»
»
»     // ***  COMPARE  **********************
»     // -- get json  (old-current | new-temp)
»     // -- compare the new temporary print output to the older saved current print
  output
»     compRestResp = util.ORCL_CN_CompareDocx(siteUrl, userAuth, processId, bsId,
  oldPrintId, newPrintId);
»
»     errVal = get(compRestResp, errorKey);
»     if(len(errVal) > 0){
»          err = "";
»          if(find(errval, "title") < 1){
»            err = errVal;
»         }else{
»                  jerr = json(errVal);
»                  err = jsonget(jerr, "title");
»         }
»         return "1~" + actionHistryV + "~"+ history +"Compare Docx Error: " + err;
»     }
»     jsonString = get(compRestResp, "jsonString");
»     if(len(get(compRestResp, "message")) > 0   OR  len(jsonString)<5){ // No
  differences found.
```

```
»         return "1~" + actionHistryV + "~"+ history +"Compare Current & Previous Docx: "
  + "No differences were found." +
»           "|" + "1~" + customerContV + "~"+ string(custContId)+
»           "|" + "1~" + previusPrintV + "~"+ string(prevPrintId)+
»           "|" + "1~" + currentPrintV + "~"+ string(curPrintId);
»     }
»     // compare succeded, differences were found
»     compRestResp = dict("string");
»
»
»     jObj = json(jsonString);
»   diffString = jsonget(jObj, "diffs");
»
»
»     // ***  MERGE  **********************
»     // -- apply compare changes to Customer Contract
»     mergeRestResp = util.ORCL_CN_MergeDocx(siteUrl, userAuth, processId, bsId,
  newPrintId, custContId, diffString);
»
»     err = "";
»     errVal = get(mergeRestResp, errorKey);
»      if(len(errVal) > 3){
»          if(find(errval, "title") < 1){
»             err = errVal;
»         }else{
»                  jerr = json(errVal);
»                  err = jsonget(jerr, "title");
»         }
»     }
»     errVal = get(mergeRestResp, "errArray");
»     if(len(errVal) > 3){
»          // pase the error array
»         jaEr = jsonarray(errVal);
»         strarray = String[jsonarraysize(jaEr)];
»         err = "\n";
»         ind = 0;
»         for e in strarray {
»             err = err + jsonarrayget(jaEr, ind) + "\n";
»             ind = ind + 1;
»         }
»     }
»     if(len(err) > 3){
»         return "1~" + actionHistryV + "~"+ history +"Merge Docx Error: " + err;
»     }
»
»     mergeBase64Doc = get(mergeRestResp, base64Key);
»     mergeRestResp = dict("string");
»
»
»
»
»
»
»     // ***  UPDATE  **********************
»     // -- if Merge successful and Document output exist, update File Attachments
»     if(len(mergeBase64Doc) > 0){
»
»         // UPDATE CUSTOMER CONTRACT
»         restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, customerContV, fileType, mergeBase64Doc,  ccdPrintFileN, true);
»         if(len(get(restResp, errorKey)) > 0){
```

```
»             return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error
  (Merged Customer Contract): " + get(restResp, errorKey);
»         }
»         mergeBase64Doc = "";
»           custContId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»
»         restResp = util.ORCL_CN_GetFileAttachmentData(siteUrl, userAuth,
  processVarName, bsId, currentPrintV);
»         currentPrintBase64 = get(restResp, base64Key);
»
»         // COPY Current Print to Previous Print
»         restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, previusPrintV, fileType, currentPrintBase64, prvPrintFileN, true);
»         if(len(get(restResp, errorKey)) > 0){
»             return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error
  (Previous Print): " + get(restResp, errorKey);
»         }
»         currentPrintBase64 = "";
»           prevPrintId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»
»         // COPY New Print to Current Print
»         restResp = util.ORCL_CN_UpdateFileAttachment(siteUrl, userAuth, processVarName,
  bsId, currentPrintV, fileType, newPrintBase64,  curPrintFileN, true);
»         if(len(get(restResp, errorKey)) > 0){
»             return "1~" + actionHistryV + "~"+ history +"Update File Attachment Error
  (Current Print): " + get(restResp, errorKey);
»         }
»         newPrintBase64 = "";
»           curPrintId = util.ORCL_CN_GetFileIdFromResponse(restResp);
»     }
»
»
»
»     return "1~" + actionHistryV + "~"+ history +"Generate & Merge Successful" +
»           "|" + "1~" + customerContV + "~"+ string(custContId)+
»           "|" + "1~" + previusPrintV + "~"+ string(prevPrintId)+
»           "|" + "1~" + currentPrintV + "~"+ string(curPrintId);
» }
```

Check For Amendments BML

```
» /*   CONTRACT NEGOTIATION  -  CHECK FOR AMENDMENTS
» *    compares Customer Contract attribute with Current Print attribute
» *    returns a list of diffrences to json string attribute
» *
» */
»
»
» siteName       = _system_company_name+".us.oracle.com";
» protocol       = "http";
» siteUrl        = protocol+"://"+siteName;
» userAuth       = "Basic " + util.getBasicAuthCredentials("cpqFullAccessUser", "User
  Authentication Error.");
»
» processId      = 36244034;       // update for commerce process
»
» curPrintSId    = ORCL_CN_GeneratedContract_t;
» custContSId    = ORCL_CN_CustomerContract_t;
»
```

```
» actionHistryV  = "ORCL_CN_ActionStatus_t";
» compareJsonV   = "ORCL_CN_CompareJSON_t";
»
» jsonReturn     = "|" + "1~" + compareJsonV + "~" ;
»
» if(len(curPrintSId) < 1){
»     return "1~" + actionHistryV + "~"+ "Compare Docx Error: " + "No document in CN -
  Current Print File Attachment." + jsonReturn + "";
» }
» if(len(custContSId) < 1){
»     return "1~" + actionHistryV + "~"+ "Compare Docx Error: " + "No document in CN -
  Customer Contract File Attachment." + jsonReturn + "";
» }
»
» // perform Compare
» restResp = util.ORCL_CN_CompareDocx(siteUrl, userAuth, processId,
  atoi(_system_buyside_id), atoi(curPrintSId), atoi(custContSId));
»
» errVal = get(restResp, "error");
» if(len(errVal) > 0){
»     jerr = json(errVal);
»     err = jsonget(jerr, "title");
»     if(len(err)<3){
»         err = errVal;
»     }
»     return "1~" + actionHistryV + "~"+ "Compare Docx Error: " + err + jsonReturn + "";
» }
» jsonString = get(restResp, "jsonString");
» if(len(get(restResp, "message")) > 0   OR  len(jsonString)<5){ // No differences found.
»     return "1~" + actionHistryV + "~"+ "Compare: No differences were found." +
  jsonReturn + "";
» }
»
»
» jsonReturn = jsonReturn + jsonString;
»
» return "1~" + actionHistryV + "~" + "Compare Completed." + jsonReturn;
```

Email Actions

Each email action should print the appropriate Email Template, and populate the Subject and Comments with the output.

Subject BML:

```
» siteName       = _system_company_name+".us.oracle.com";
» protocol       = "http";
» siteUrl        = protocol+"://"+siteName;
»
» userAuth       = "Basic " + util.getBasicAuthCredentials("cpqFullAccessUser", "User
  Authentication Error.");
»
» processVarName = "oraclecpqo";
» bsId           = atoi(_system_buyside_id);
» templateLang   = -1;
» templateName   = "ORCL_CN Send to Customer Template";
»
»
» restResp = util.ORCL_CN_PrintEmailDesigner(siteUrl, userAuth, processVarName, bsId,
  templateName, templateLang);
»
```

```
» subject = get(restResp, "subject");
» if(len(subject) == 0){
»   subject = "This is your new Contract.";
» }
» return subject;
```

Comments BML:

```
» siteName       = _system_company_name+".us.oracle.com";
» protocol       = "http";
» siteUrl        = protocol+"://"+siteName;
»
» userAuth       = "Basic " + util.getBasicAuthCredentials("cpqFullAccessUser", "User
  Authentication Error.");
»
» processVarName = "oraclecpqo";
» bsId           = atoi(_system_buyside_id);
» templateLang   = -1;
» templateName   = "ORCL_CN Send to Customer Template";
»
»
» restResp = util.ORCL_CN_PrintEmailDesigner(siteUrl, userAuth, processVarName, bsId,
  templateName, templateLang);
»
» body = get(restResp, "body");
» if(len(body) == 0){
»     body = "Your Contract is attached. Please respond with any changes in a modified
  attachment. Thank you.";
» }else{
»     body = replace(body, "\\n", "");
» }
» return body;
```

Be sure to replace the Customer Template name with the Legal Department Template name in the “Send to Legal” Email Action.

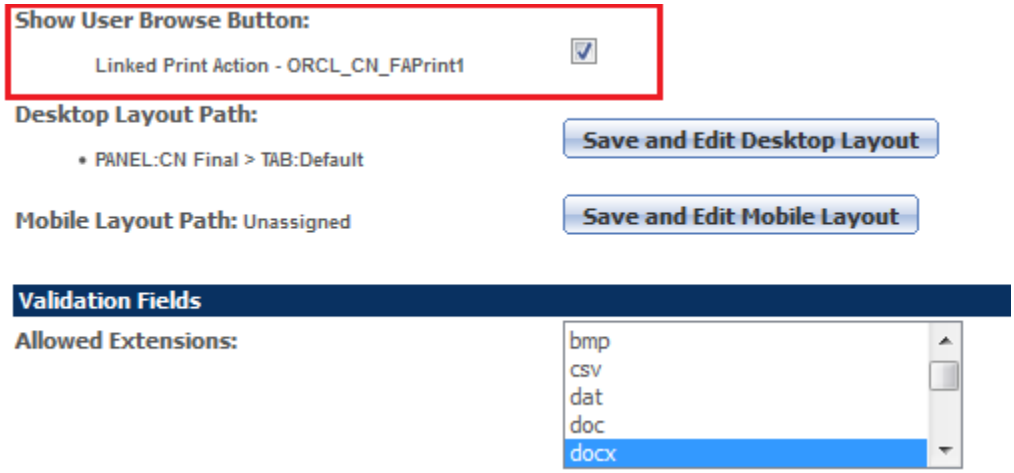
Advanced Validation Rule for both Email Actions when Customer Contract is not populated:

```
» if(len(ORCL_CN_CustomerContract_t)==0){
»   return "Customer Contract is empty.";
» }
» return "";
```

Select “Customer Contract” as an Attachment under “Additional Attachments”.

Attributes Setup

Each File Attachment attribute should be DOCX only. The attributes should be associated with a **Print Action**, **Saved Automatically**, and contain a **Browse** button (use ORCL_CN_FAPrint1 -4).



File Attachment Setup

The Amendments Display Default Value should be the result of a function:

```
» return commerce.ORCL_CN_ProcessJsonToTable(ORCL_CN_CompareJSON_t);
```

Process JSON to Table BML

```
» diffSize = 0;
» diffArray = jsonarray("[]");
» // turn json string to json object
» if(find(jsonInput, "diffs")>0){
»   jobj = json(jsonInput);
»   str = jsonget(jobj, "diffs");
»   print("str");
»   print(str);
»
»   jary = jsonarray(str);
»   print("DOBJ");
»   print(jary);
»
»   diffSize = jsonarraysize(jary);
»   diffArray = jary;
» }
»
» if(diffSize < 1){
»   return "";
» }
»
» // table html builder
» tableString = "<table class='cn_compare_table'>";
»
»  // HEADING ROW
» tableString = tableString +
» "<tr>" +
» //"<td>Clause Id</td>" +
```

```
» //"<td>Action Type</td>" +
» "<th>Clause</th>" +
» "<th>Section</th>" +
» "<th>Reviewed</th>" +
» "<th>Action</th>" +
» "</tr>";
»
» rowCounter = 2;
» index = 0;
» strArray = String[diffSize];
» // for each diff returned in json
» for  str in strArray   {
»   diffStr = jsonarrayget(diffArray, index);
»   diffObj = json(diffStr);
»
»   inserted = false; updated = false; removed = false; moved = false;
»
»   // Clause Label
»   clauseLabel = trim(jsonget(diffObj, "clauseLabel"));
»   clauseLabel = replace(clauseLabel, "&", "&#38;");
»   clauseLabel = replace(clauseLabel, "<", "&#60;");
»   clauseLabel = replace(clauseLabel, ">", "&#62;");
»
»   // Parent Clause Label
»   parentLabel = trim(jsonget(diffObj, "parentLabel"));
»   parentLabel = replace(parentLabel, "&", "&#38;");
»   parentLabel = replace(parentLabel, "<", "&#60;");
»   parentLabel = replace(parentLabel, ">", "&#62;");
»
»   // Are Changes Reviewed?
»   reviewed = trim(jsonget(diffObj, "reviewed"));
»
»   // Change Type Label
»   action = trim(jsonget(diffObj, "action"));
»   if(find(action, "Inserted")>-1){
»           inserted = true;
»   }elif(find(action, "Updated")>-1){
»           updated = true;
»   }elif(find(action, "Removed")>-1){
»           removed = true;
»   }elif(find(action, "Moved")>-1){
»           moved = true;
»   }
»
»
»   cellString = "<td>"+clauseLabel+"</td>" + "<td>"+parentLabel+"</td>" +
  "<td>"+reviewed+"</td>" + "<td>"+action+"</td>";
»
»   claz = "bgcolor-list-even"; if(rowCounter % 2 <> 0) {claz = "bgcolor-list-odd";}
»   if(inserted) {claz = claz+" "+"insert";}       if(updated) {claz = claz+"
  "+"update";}
»   if(removed) {claz = claz+" "+"remove";}               if(moved) {claz = claz+"
  "+"move";}
»
»   tableString = tableString + "<tr class='"+claz+"'>" + cellString + "</tr>";
»
»   rowCounter = rowCounter + 1;
»   index = index + 1;
» }
»
» tableString = tableString + "</table>";
```

```
»
» return tableString;
```

BML Library Scripts Setup

BML Library Scripts

Label	Input Parameters	Description
ORCL_CN_GetFileAttachmentData	siteUrl String userAuth String processVarName String transactionId Integer attachmentVarName String	String Dictionary - XSLX Response, will include the "base64doc" key.
ORCL_CN_UpdateFileAttachment	siteUrl String userAuth String processVarName String transactionId Integer attachmentVarName String fileType String fileContent String fileName String hasBrowse Boolean	String Dictionary - Successful response will include "fileId".
ORCL_CN_GetFileIdFromResponse	response String Dictionary	Integer - The File Id found in the response or "-1".
ORCL_CN_PrintEmailDesigner	siteUrl String userAuth String processVarName String transactionId Integer templateName String templateLanguage Integer	String Dictionary - Returns "error" or "subject" & "body".
ORCL_CN_PrintDocDesigner	siteUrl String userAuth String processVarName String transactionId Integer templateName String templateLanguage Integer outputFormat String	String Dictionary - Returns "error" or "base64doc".

ORCL_CN_CompareDocx	siteUrl String userAuth String processId Integer transactionId Integer prevFileAttachId Integer curFileAttachId Integer	String Dictionary - Returns "jsonString", "message" or "error".
ORCL_CN_MergeDocx	siteUrl String userAuth String processId String transactionId Integer sourceFileId Integer targetFileId Integer compareJsonString String	String Dictionary - Response includes "base64doc" of modified Target document, or "error", or "errArray".
getBasicAuthCredentials	accountName String defaultReturnValue String	String - Gets base64 encoded credentials from the secure data column table "Credentials"

Get File Attachment Data BML

```
» response = dict("string");
»
» url = siteUrl+"/rest/v2"+
» "/commerceProcesses/"+ processVarName+
» "/transactions/"+ string(transactionId)+
» "/attachments/"+ attachmentVarName+"/";
»
» headers = dict("string");
» put(headers, "Authorization", userAuth );
» put(headers, "Accept", "application/json");
»
» // call rest method
» xlsxResponse = urldata( url, "GET", headers );
» //print(xlsxResponse);
»
» if( get(xlsxResponse, "Status-Code") <> "200" ){
»     put(response, "error", get(xlsxResponse, "Error-Message"));
»     return response;
»      /* ************************************ */
»
» }else{
»     mb = get(xlsxResponse, "Message-Body");
»     jmb = json(mb);
»     fileContent = jsonget(jmb, "fileContent");
»     if( len(fileContent) < 5 ){
»         put(response, "error", "GetFileAttachment: Could not locate document in
  Response.");
»         return response;
»     }
»     put(response, "base64doc", fileContent);
»     return response;
» }
```

Update File Attachment BML

```
» response = dict("string");
»
» url = siteUrl+"/rest/v2"+
»     "/commerceProcesses/" + processVarName+
»   "/transactions/"      + string(transactionId) +
»   "/attachments/"       + attachmentVarname;
»
» headers = dict("string");
» put(headers, "Authorization", userAuth );
» put(headers, "Accept", "application/json");
» put(headers, "Content-Type", "application/json");
»
» // add json data
» usePrintContext = "false";
» if(NOT hasBrowse){
»     usePrintContext = "true";
» }
»
» jsonObj = json();
» jsonput(jsonObj, "mediaType", fileType);
» jsonput(jsonObj, "fileContent", fileContent);
» jsonput(jsonObj, "useTempTable", usePrintContext);
» jsonput(jsonObj, "usePrintContext", usePrintContext);
» jsonput(jsonObj, "fileName", fileName);
»
» // call rest method
» xlsxResponse = urldata( url, "POST", headers, jsontostr(jsonObj) );
»
» if( get(xlsxResponse, "Status-Code") <> "200" ){
»     put(response, "error", get(xlsxResponse, "Error-Message"));
»     return response;
» }else{
»     return xlsxResponse;
» }
```

Get File Id From Response BML

```
» mb = get(response, "Message-Body");
»
» jmb = json(mb);
» ids = jsonget(jmb, "fileId");
»
» if( isnumber(ids) ){
»     return atoi(ids);
» }
»
» return -1;
```

Print Email Designer BML

```
» // Prints a Email Designer Template. Returns "error" or "subject" & "body".
»
» response = dict("string");
»
» /* ********************************** */
» // PRINT TEMPLATE
```

```
» url = siteUrl+"/rest/v2/emailGenerator/";
»
» headers = dict("string");
» put(headers, "Authorization", userAuth );
» put(headers, "Accept", "application/json");
» put(headers, "Content-Type", "application/json");
»
» // add json data
» jsonObj = json();
» jsonput(jsonObj, "processVarname", processVarName);
» jsonput(jsonObj, "templateName", templateName);
» jsonput(jsonObj, "transactionId", transactionId);
» jsonput(jsonObj, "languageCode", templateLang);
»
» // call rest method
» xlsxResponse = urldata( url, "POST", headers, jsontostr(jsonObj) );
»
» // CHECK OUTPUT, UPDATE STATUS
» statusCode = get(xlsxResponse, "Status-Code");
» printStatus = "";
» if( statusCode <> "204" ){
»     printStatus = "Print FAILED: " + templateName + "\n" +
»     "Error: " + get(xlsxResponse, "Error-Message");
»
»     put(response, "error", printStatus);
»     return response;
»     /* ************************************ */
»
» }else{
»     printStatus = "Print Completed: " + templateName + "\n";
» }
»
» /* ************************************ */
»
» // GET PRINT OUTPUT
» url = get(xlsxResponse, "Location");
»
» headers = dict("string");
» put(headers, "Authorization", userAuth);
» put(headers, "Accept", "application/json");
»
» // get print output call
» xlsxResponse = urldata( url, "GET", headers );
» mb = get(xlsxResponse, "Message-Body");
»
» if(len(mb) > 0){
»     jmb = json(mb);
»
»     printOutput = jsonget(jmb , "subject" );
»     put(response, "subject", printOutput);
»
»     printOutput = jsonget(jmb , "body" );
»     put(response, "body", printOutput);
» }else{
»     put(response, "error", "Email Output was empty.");
» }
»
» print("PRINT COMPLETE");
» print("");
»
» return response;
```

Print Doc Designer BML

```
» // Prints a Document Designer Document. Returns "error" or "base64doc".
»
» response = dict("string");
»
» /* ************************************ */
» // PRINT TEMPLATE
» url = siteUrl+"/rest/v2/documentGenerator/";
»
» headers = dict("string");
» put(headers, "Authorization", userAuth );
» put(headers, "Accept", "application/json");
» put(headers, "Content-Type", "application/json");
»
» // add json data
» jsonObj = json();
» jsonput(jsonObj, "processVarname", processVarName);
» jsonput(jsonObj, "templateName", templateName);
» jsonput(jsonObj, "transactionId", transactionId);
» jsonput(jsonObj, "languageCode", templateLang);
» jsonput(jsonObj, "outputFormat", outputFormat);
»
» // call rest method
» xlsxResponse = urldata( url, "POST", headers, jsontostr(jsonObj) );
»
» /* ************************************ */
»
» // CHECK OUTPUT, UPDATE STATUS
» statusCode = get(xlsxResponse, "Status-Code");
» printStatus = "";
» if( statusCode <> "204" ){
»     printStatus = "Print FAILED: " + templateName + "\n" +
»     "Error: " + get(xlsxResponse, "Error-Message");
»
»     put(response, "error", printStatus);
»     return response;
»     /* ************************************ */
»
» }else{
»     printStatus = "Print Completed: " + templateName + "\n";
» }
»
» /* ************************************ */
»
» // GET PRINT OUTPUT
» url = get(xlsxResponse, "Location");
»
» headers = dict("string");
» put(headers, "Authorization", userAuth);
» put(headers, "Accept", "application/json");
»
» // get print output call
» xlsxResponse = urldata( url, "GET", headers );
» mb = get(xlsxResponse, "Message-Body");
»
» if(len(mb) > 0){
»     jmb = json(mb);
»     printOutput = jsonget(jmb , "document" );
```

```
»     put(response, "base64doc", printOutput);
» }else{
»     put(response, "error", "Document Output was empty.");
» }
»
» print("PRINT COMPLETE");
» print("");
»
» return response;
```

Compare DOCX BML

```
» response = dict("string");
»
» // GENERATE DIFF REST CALL
» url = siteUrl+"/rest/v2/docxCompare”;
»
» jsonObj = json();
» jsonput(jsonObj, "processId", processId);
» jsonput(jsonObj, "transactionId", bs_id);
» jsonput(jsonObj, "oldFileAttachId", prevFileAttachId);
» jsonput(jsonObj, "newFileAttachId", curFileAttachId);
»
» headers = dict("string");
» put(headers, "Authorization", userAuth );
» put(headers, "Accept", "application/json");
» put(headers, "Content-Type", "application/json");
»
» // call rest method
» xlsxResponse = urldata( url, "POST", headers, jsontostr(jsonObj) );
»
» if( get(xlsxResponse, "Status-Code") <> "200" ){
»     err = get(xlsxResponse, "Error-Message");
»     if(len(err) <= 5){
»         err = get(xlsxResponse, "Message-Body");
»     }
»     put(response, "error", err);
»     return response;
»
» }else{
»     mb = get(xlsxResponse, "Message-Body");
»     jmb = json(mb);
»
»     diffs = jsonget(jmb , "diffs" );
»
»     if(len(diffs) > 3){
»         put(response, "jsonString", mb);
»     }
» }
»
» print("COMPARE COMPLETE");
» return response;
```

Merge DOCX BML

```
» response = dict("string");
»
» // GENERATE MERGE REST CALL
» url = siteUrl+"/rest/v2/docxMerge";
»
```

```
» jsonObj = json();
» jsonput(jsonObj, "processId", processId);
» jsonput(jsonObj, "transactionId", bs_id);
» jsonput(jsonObj, "sourceFileAttachId", sourceFileId);
» jsonput(jsonObj, "targetFileAttachId", targetFileId);
» jsonput(jsonObj, "approvedChanges", jsonarray(compareJsonString));
»
» headers = dict("string");
» put(headers, "Authorization", userAuth );
» put(headers, "Accept", "application/json");
» put(headers, "Content-Type", "application/json");
»
» // call rest method
» xlsxResponse = urldata( url, "POST", headers, jsontostr(jsonObj) );
»
» if( get(xlsxResponse, "Status-Code") <> "200" ){
»     er = get(xlsxResponse, "Error-Message");
»     if(len(er) <= 5){
»         er = get(xlsxResponse, "Message-Body");
»     }
»     put(response, "error", er );
»     return response;
»
» }else{
»     mb = get(xlsxResponse, "Message-Body");
»     jmb = json(mb);
»
»
»     errors = jsonget(jmb , "errors" );
»     if(len(errors) > 5){
»         put(response, "errArray", errors);
»         print("MERGE COMPLETE WITH ERRORS");
»         return response;
»     }
»
»     document = jsonget(jmb , "base64Doc" );
»     put(response, "base64doc", document);
»     print("MERGE COMPLETE");
»     return response;
» }
```

Get Basic Auth Credentials BML

```
» //get credentials for JCS-SXs 2 legged OAuth client that was stored by you in secure
  columns of custom data tables
» clientname = accountName;
» clients = bmql("select username, password from Credentials where
  integrationName=$clientname");
» //base 64 encode the credentials, per Basic Authorization standard
» for client in clients {
»   encodedCreds= encodebase64( get(client, "username") + ":" + get(client, "password"));
»   return encodedCreds;
» }
» //allow appropriate message to user, if integration client information was not located
» return defaultReturnValue;
```

Data Table Setup

Credentials

Schema

Name	Type	Description
integrationName	String	Client name.
username	Secure	Encoded user name.
password	Secure	Encoded user password.
desc	String	Description

Commerce Layout Setup

Add the Customer Contract File Attachment Attribute, the Generate Action, the Amendments Display Attribute and the Check for Amendments Action to a new Layout Panel on the Quote level Layout. Add the Email To Customer Action and the Email to Legal Action. Any other attributes that you might need for debugging, such as the Compare JSON Attribute or Status Update Attribute can be added as well.

Template Setup

Create the Email Templates that will be used to inform customers or legal that their attention is required. The emails may include the instruction to “update the Contract using Microsoft Word and reply with the modified file attached.” Create the Document Designer Template with any necessary Sections and Elements used for the Negotiated Contract. Be sure to select the “Contract” option. The Sections and Elements can be renamed to designate specific areas as recognizable clauses.

Ready To Go

After deploying the Templates, Rules, and Commerce Process, the Contract Negotiation process should be ready to go. Use the Sample Flow description to test the setup and make any adjustments required by your business use cases.

Conclusion

A custom Contract Negotiation solution is made available through proper setup of CPQ Commerce Actions and Attribute that take advantage of new functionality provided by the Compare and Merge REST methods.

Oracle Corporation, World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065, USA

Worldwide Inquiries
Phone: +1.650.506.7000
Fax: +1.650.506.7200

CONNECT WITH US

blogs.oracle.com/oracle
facebook.com/oracle
twitter.com/oracle
oracle.com

Hardware and Software, Engineered to Work Together

Contract Negotiation
July 2023