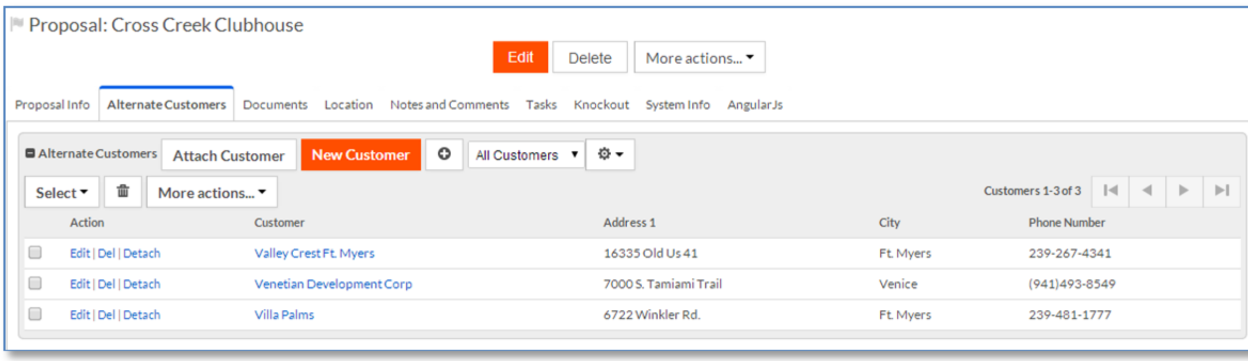
Email Templates – Part 2

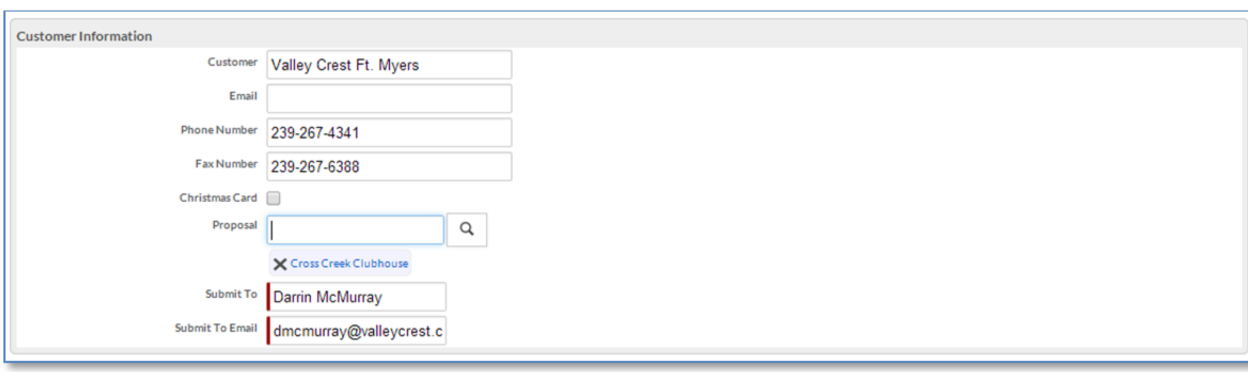
Creating Email Templates and sending to multiple related customers.

As we saw in an earlier tutorial, we can have a predefined email template get sent to a single entity. In our case the entity was the contact on the original opportunity. When that opportunity was converted to a proposal, it automatically sent the email to the address specified.

Per the specifications, once a proposal gets created, it may be assigned to MULTIPLE customers, as multiple contractors are all bidding on the same contract. As such, once the take-off is done, and the items are priced, it's desirable to click one button and have the emails automatically sent to all related customers.



In order for this to work, each related customer MUST have a submit to and submit to email field defined. As such, those fields are marked as required as shown here.

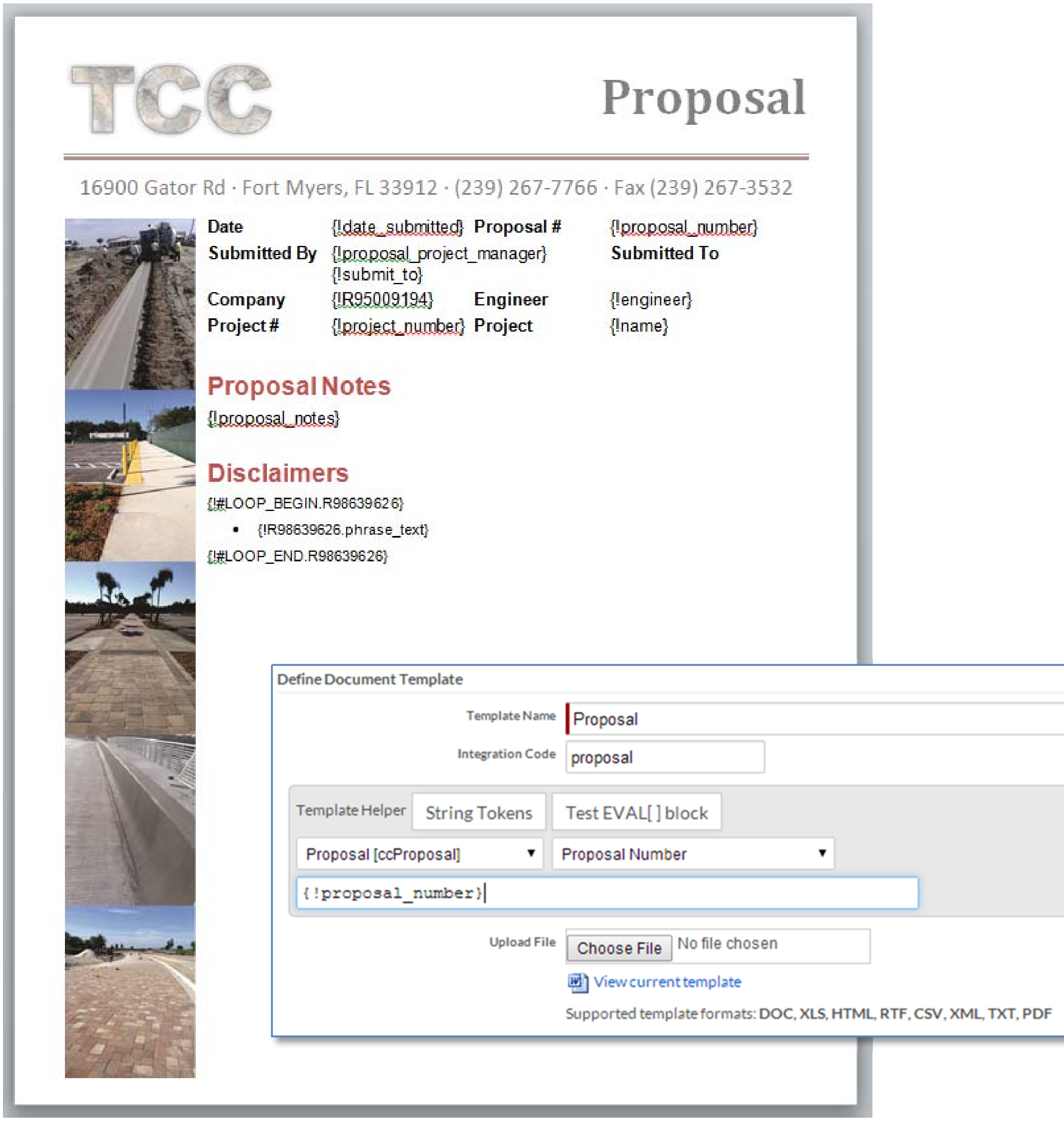


Creating the Word Template

The next thing we need is an email template to send. This can be a PDF, Word Document, or even a spreadsheet. The complete list can be found in the Rollbase in Action documentation. For this tutorial, we'll simply use a Word document. When the application goes to production, we'll convert that document into a PDF so it reaches a larger audience. We don't want to require our customers to have

Microsoft Word to be able to read our proposal. Important note regarding Word and Excel, you MUST save in version 2003 or earlier (.doc, .xls). The newer versions are not currently supported.

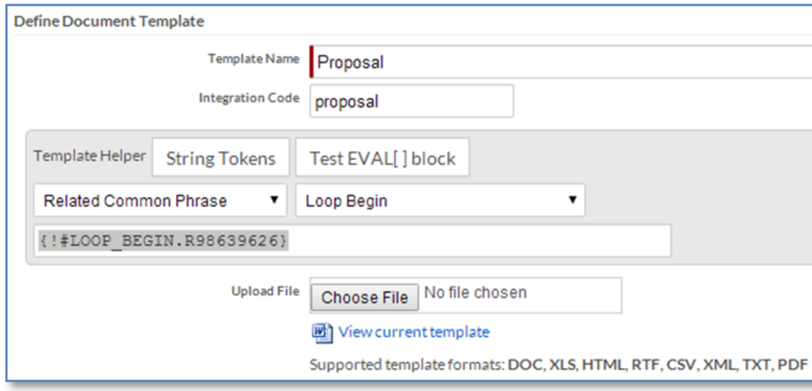
We have created a multi-page document and included the merge codes. The merge codes can be found when editing your document template.



On the first page of this proposal, we have included some images, general header information, and the proposal notes, all of which are singular fields that can be found in the template helper shown above. Additionally, we have a related table of Disclaimers which are assigned to the proposal by the project manager. We want them to display as bullet points on the first page, so we introduce a new document merge feature for loops.

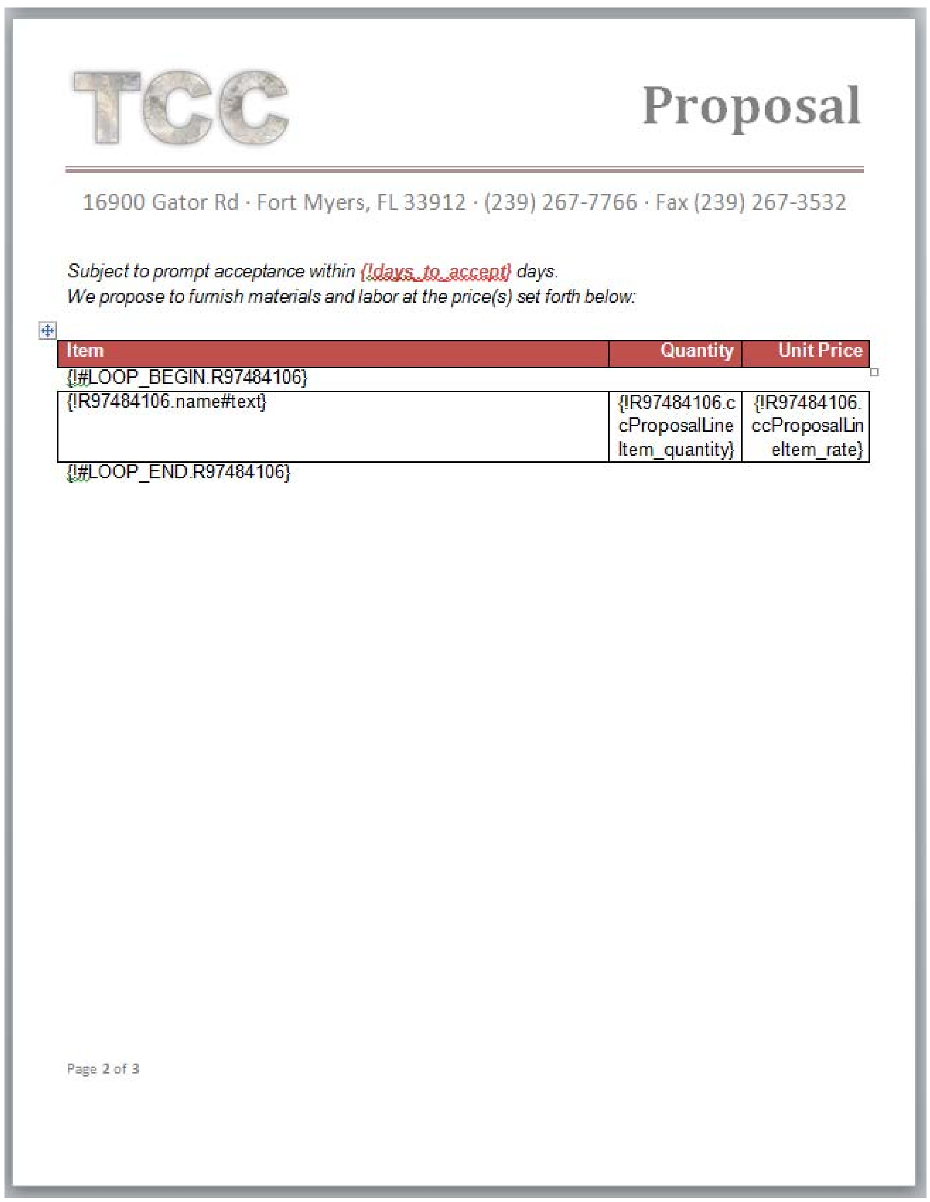
```
{!#LOOP_BEGIN.R98639626}
  • {!R98639626.phrase_text}
{!#LOOP_END.R98639626}
```

The Template Helper will give you the token codes. We just need the loop begin and loop end. Whatever we put inbetween will get duplicated for each related record. We simply want a bullet and the disclaimer.



The second page of the proposal is another loop through related proposal line items. The technique is the same as above, but we've enhanced it by putting them in a table.

The last page of the proposal is simply static fields from the proposal, so we won't show that in this document.



TCC Proposal

16900 Gator Rd · Fort Myers, FL 33912 · (239) 267-7766 · Fax (239) 267-3532

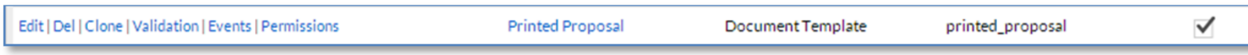
Subject to prompt acceptance within {!days_to_accept} days.
We propose to furnish materials and labor at the price(s) set forth below:

Item	Quantity	Unit Price
{!#LOOP_BEGIN.R97484106}		
{!R97484106.name#text}	{!R97484106.cProposalLineItem_quantity}	{!R97484106.ccProposalLineItem_rate}
{!#LOOP_END.R97484106}		

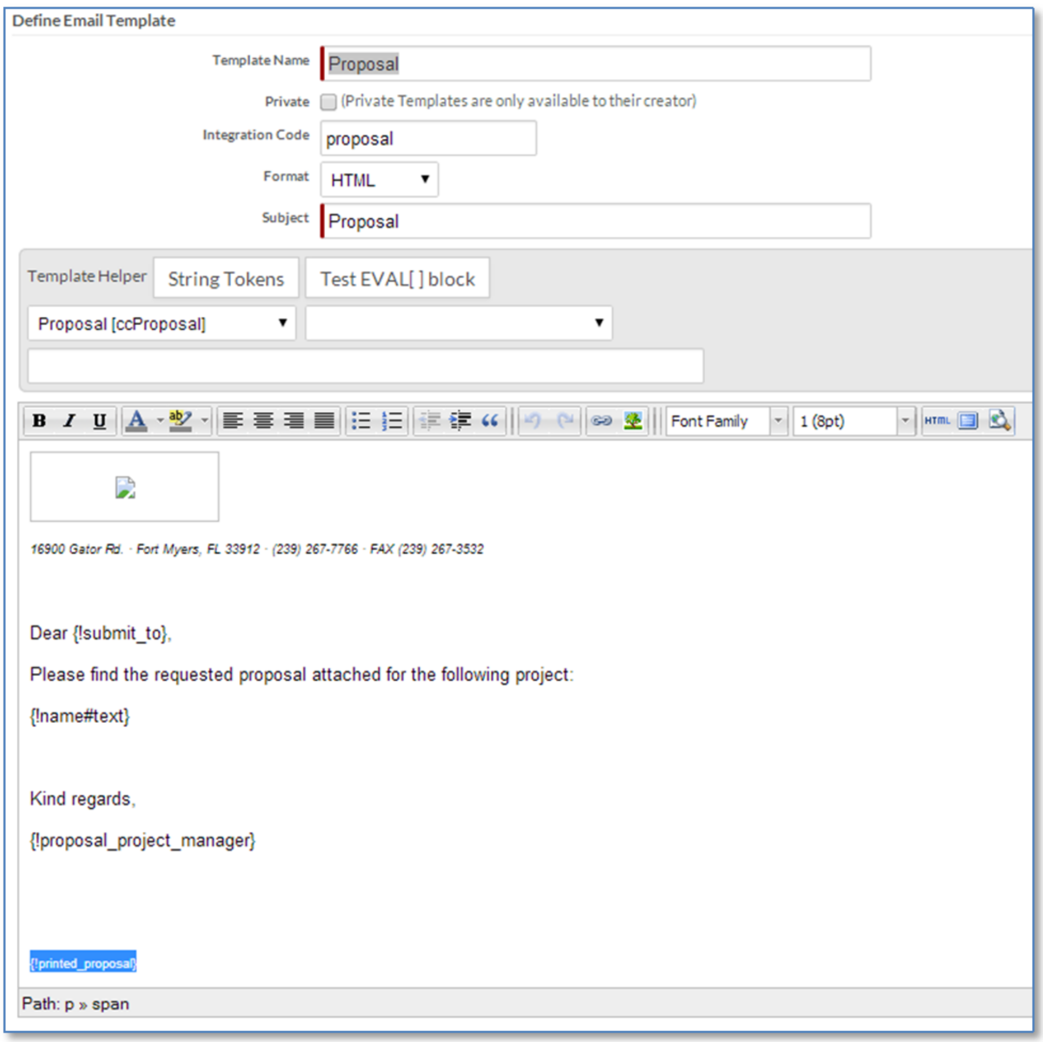
Page 2 of 3

Creating the Email Template

While you can email the Word document template directly to the customer, we need an HTML email template AND a document template field within the proposal. The template field also gets the default value upon creation pointing to the document that we want.



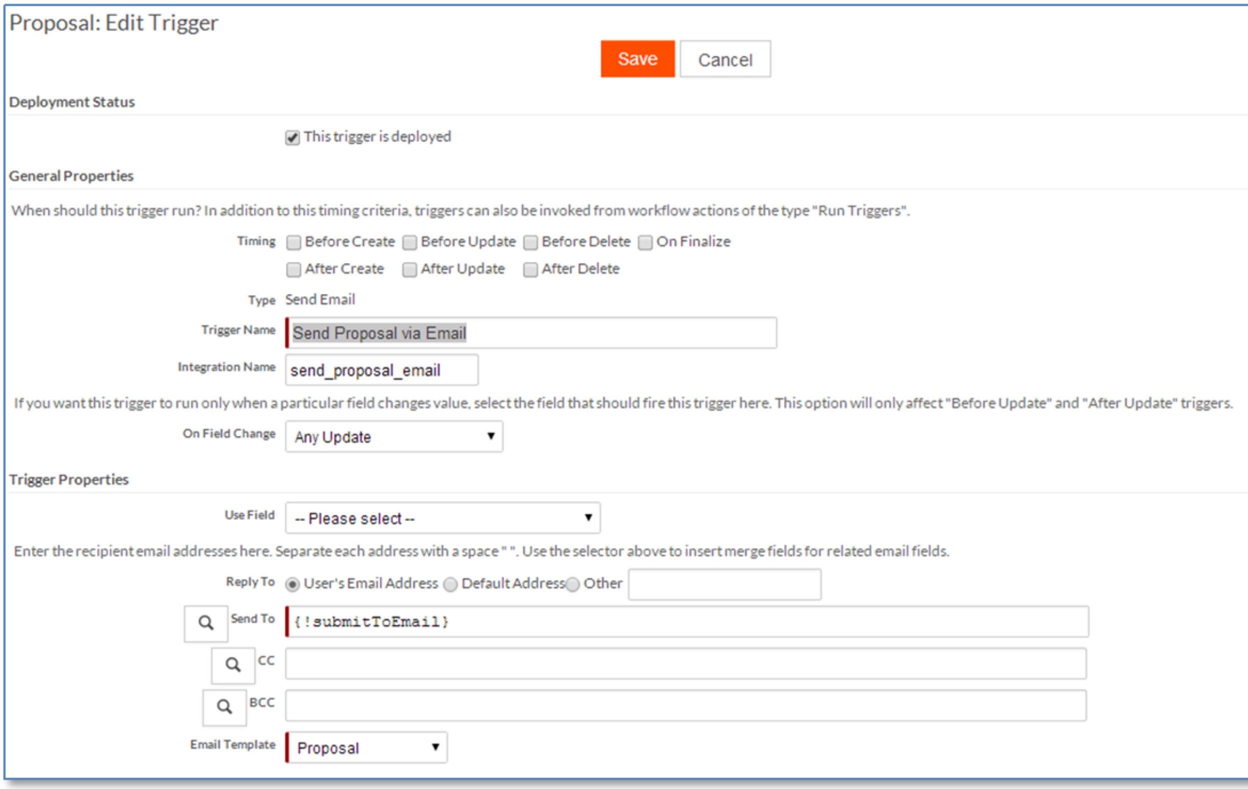
The important thing to take away from this is that if you put the TEMPLATE TOKEN for the document template field. By doing this, Rollbase will ATTACH the document to the HTML email template when it gets sent.



You will notice I have put the {!printed_proposal} token at the bottom of the template. The text is considerably smaller and WHITE in color. This is because you will actually see a merge value in the email sent. This is just an attempt to make it more obscure, but it WILL attach the word document to the email.

Creating the triggers

We actually need TWO triggers to pull this off. The first trigger is of type "Send Email". In here, simply specify the {!submitToEmail} field from the proposal and select the email template (word doc).



The second trigger is an Object Script Trigger that's a bit more involved. This trigger loops through related customers, grabs their submit to and submit to email fields and updates the proposal. Then it calls the email trigger to send for the current record. It repeats this until all related customers have been sent. The code and explanation follows.

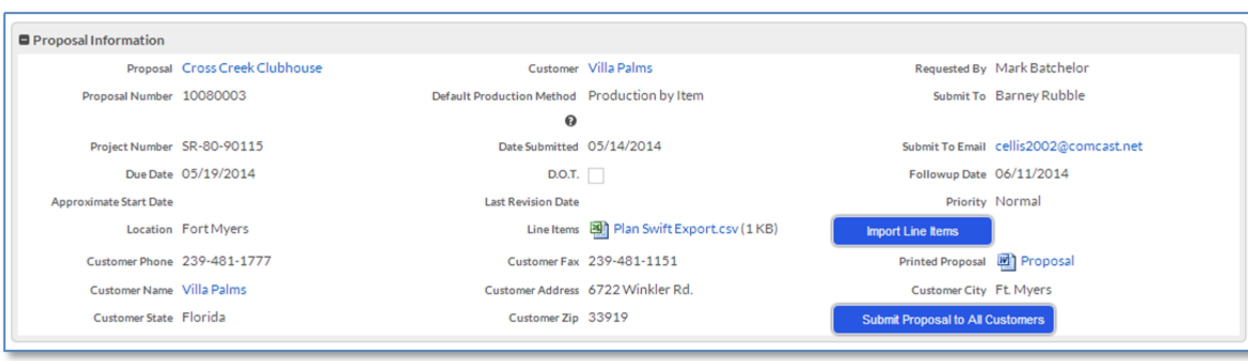
```
//loop through the related customers and set each one current and send proposal.
//Store the proposal id to a variable
var proposalId = new Number("{!id}");
//Get the value of the current date.
var today = new Date(rbv_api.getCurrentDate());
//Create an array that we can put this date.
var propArr = new Array();
//The proposal field we want to update is the date_submitted.
propArr["date_submitted"] = today;
//Update the proposal record.
rbv_api.updateRecord("ccProposal", {!id}, propArr);
//Create another array to hold the information from the related customers to be put to
//the proposal.
var arr;
//Start the loop on related customers.
{!#LOOP_BEGIN.R95925946}
  arr = new Array();
```

```
  //Copy the related customer fields to the array.
  arr["submit_to"] = "{!R95925946.submit_to}";
  arr["submitToEmail"] = "{!R95925946.submitToEmail}";
  arr["R95009194"] = parseInt("{!R95925946.id}");
  //Set the today variable to 2 weeks from today.
  today.setDate(today.getDate()+14);
  //Update the followup_date field in the aray.
  arr["followup_date"] = today;
  //Update the proposal record. Note we use the proposalId instead of the {!id} token.
  //that token seems to work OUTSIDE of the loop, but I had issues inside it.
  rbv_api.updateRecord("ccProposal", proposalId , arr);
  //Call the send email trigger.
  rbv_api.runTrigger("ccProposal", proposalId , "send_proposal_email", false);
  //The last thing this trigger is doing is creating a new record in the proposal
  //tasks object. We want one task per related customer. These tasks will integrate
  //into the Rollbase calendar interface.
  var arr2 = new Array();
  arr2["assignedTo5"] = {!proposal_project_manager#id};
  arr2["description"] = "{!project_name} - 14 Day followup with
{!R95925946.submit_to}";
  arr2["dueDate"] = today;
  arr2["taskSubject"] = "{!project_name} - 14 Day followup with
{!R95925946.submit_to}";
  arr2["priority"] = {!proposal_priority#id};
  arr2["R95911408"] = proposalId ;
  rbv_api.createRecord("ccProposal_task",arr2);

{!#LOOP_END.R95925946}
```

Adding a button to the UI and calling the trigger

Like other buttons we have placed on the UI in previous lessons, I'll skip coding for this button, but you will notice that I put a button "Submit Proposal to All Customers" on the proposal screen. This is done through a script component and uses the same type of code to call a trigger as we've documented before.



The net result of all of this effort is that a copy of the proposal gets sent to each related customer as expected, with their information merged. It also creates the follow-up tasks for each customer for the project manager to maintain.