



Service Manager Smart Indicator

How to create and modify Smart Indicator Functionality



Introduction.....	2
Prerequisites.....	2
Creating a Smart Indicator.....	3
Creating the Context Definition record.....	3
Creating the scmessage record.....	3
Activating the Smart Indicator dynamically.....	4
Validating Data Policy.....	4
Creating the Smart Indicator button.....	5
Adding to an Existing Smart Indicator.....	7
Creating a Context Action.....	7
Creating a new context wizard.....	8
For more information.....	10

Introduction

The Service Manager Smart Indicator provides information on related issues and the ability to drill down to this data quickly in order to troubleshoot the service user's issue. Smart Indicators can be configured to show other interactions and incidents open and closed for the same service user, incidents associated with the same service or incidents associated with the same configuration item (CI). Smart Indicators can also be configured to show known errors for a service thus providing the ability to communicate to the caller information related with the service immediately.

This document shows how to create both a Smart Indicator on the Service field within Change Management that will show incidents and known errors for this service that may affect how or when the planned change will be implemented, as well as how to search for open changes for a Service within an Interaction.

Prerequisites

- Service Manager 7.01 or higher
- Experience with the Service Manager tailoring tools
- Good knowledge of Service Manager JavaScript

Creating a Smart Indicator

To create the Smart Indicator button on a new form, you first have to create the Context Definition record if one does not already exist. A new Display Option that is executed when clicking the button will also need to be created and the State record will need to be edited to call the Process that will perform the action. Finally a new button will need to be added to the form. Additional configuration steps are also necessary to determine if data is available for the item in order to activate or deactivate the Smart Indicator button.

Creating the Context Definition record

Ensure that there is a Context Definition record for the table and field for which the Smart Indicator is being activated. You can find this information by going to the contextDefinition table in Database Manager. For this example, create a new record for the table cm3r and field affected.item with 3 defined actions that will return data if it exists, as shown in the picture below. The out-of-the-box system has 9 pre-defined Context Actions.

The screenshot shows a software interface with a toolbar at the top containing icons for OK, Cancel, Previous, Next, Add, Save, Delete, Find, and Fill. Below the toolbar is a message box that says "Context Definition record added." The main form is titled "Context Definition" and contains the following fields:

- Table: cm3r
- Field: affected.item
- State: (empty dropdown)
- Profile: (empty dropdown)

Action Name	Action Condition
Open Incidents for this Service	true
Closed Incidents for this Service	true
Possible workarounds for this Service	true

Creating the scmessage record

Ensure that there is a scmessage record for all actions that were defined in the contextDefinition record above. In Database Manager, search on the scmessage table for a Class of contextAction. If there is no scmessage record for an action, add a record to the scmessage table with the Action Name from the Context Definition record above in both the Message Number and Text fields.

HP Service Manager Message

Language Code:	en
Class:	contextAction
Message Number:	Open Incidents for this Servic
Severity:	5
Text:	Open Incidents for this Service
Comments:	

Activating the Smart Indicator dynamically

In order for the Smart Indicator to activate indicating that there is data related to a particular service, format control calculations must be added to the cm3r format control record. There are 8 lines which may be copied from the incidents format control calculations.

Initial = true

```
$L.void=jscall("context.ResetForFields", $file)
```

Display = true

```
$context.contextExists=nullsub($contextExists,  
true);$context.field.name=nullsub($field.name, "")
```

```
$context.profile.name=nullsub($profile.name,  
"");$context.state.name=nullsub($state.name, "")
```

```
cleanup($contextExists);cleanup($field.name);cleanup($profile.name);clean  
up($state.name)
```

initial = true

```
$contextExists=false;cleanup($field.name);cleanup($context.state)
```

Display = \$context.contextExists

```
$L.context.result=jscall("context.GetResult", $file, filename($file),  
$context.field.name, $context.state.name, $context.profile.name, true)
```

```
$L.eval.string="if ($L.context.result>0) then  
($"+str($context.field.name)+".on=\"true\") else  
($"+str($context.field.name)+".on=\"false\")"
```

```
$L.void=evaluate(parse($L.eval.string, 11))
```

Validating Data Policy

Verify that within Data Policy the field associated with the Smart Indicator, in this case affected.item, has the Usage Type defined as Application or Data. If it is defined as System or Deprecated, it will not activate as the JavaScript is coded to ignore these Usage Types.

Data Policy

Name:

SQL Base Name:

Unique Key:

Default Format:

Prohibit Default Access

System Table?

Description:

General | Da

Applications:

Area:

Record ID:

Field Name	Caption	Field Type	Usage Type	Availab
affected.item	Service		Application	true

Creating the Smart Indicator button

Determine the form where the Smart Indicator will be placed and the button id that will be assigned to the action. To determine which option is available, go to the related display screen, search for all options used on that screen and sort by option number. For the purpose of this implementation, choose a number between 200 and 2000. In our example we will be using the CM.change.logging form and a button id of 399 which will be added to the cm.view.display screen within the Display Option module.

Display Application Option Definition

Screen ID: Modifies Record Action:

Unique ID: back, close, and more are special

GUI option: Balloon Help (If Option < 200):

Text Option: Default Label:

Bank: Text Alternative:



Condition:



User Condition:

To execute the smart action when the button is clicked, the runcontext Action will need to be added to the State Definition record with the Process Name of run.context.wizard and a Condition of true.

State Definition

State:

Display Screen:  

Initialization Process:  

Format:

Input Condition (view state only):

Non-base methods

Display Action	Process Name	Condition	Save First
reject	cm.reject	true	
withdraw	cm.withdraw	true	
runcontext	run.context.wizard	true	

Finally, add a button to the CM.change.logging form next to the Service field with the following values.

The screenshot shows the Forms Designer interface with a form titled 'Change Data' and a Properties window for a 'Button' control. The form contains several input fields, and a button is being added next to the 'Service' field. The Properties window shows the following configuration for the button:

Property	Value
Name	button1210606473958
X	70
Y	26
Width	5
Height	2
Visible	<input checked="" type="checkbox"/>
Visible Condition	
Category	No_Context
Caption Condition	
Input	\$\$affected.item
Tab Stop	-1
Justification	Left
Button ID	399
Button ID Condition	
Bitmap File	no_context
Bitmap Condition	[\$affected.item.on] ? "true":context, no_context
Balloon Help	

Note: The input value must be a variable and the same name as the input field with which it is associated.

Adding to an Existing Smart Indicator

To add a new search to an existing Smart Indicator you will need to create a new Context Action and then add it to a Context Definition record.

Creating a Context Action

In this example, a new Context Action to search for open changes against a service will be created. This is done by accessing the contextAction table within Database Manager. The code will be similar to the existing Context Actions so the most efficient method will be to modify the necessary fields in, for example, "Open Incidents for this Service" and choose to "Add" the new Context Action. The modifications needed are the Name, Description is optional, and within the Code change "probsummary" to "cm3r" and flag = true to open = true. Choose Add to save the new Context Action.

Context Action

Name:

Description:

Code:

```
1 var fRecords = new SCFile("cm3r");
2 var sql = "open=true and affected.item=\""+fFile.affected_item+"\"";
3 rc = fRecords.doSelect( sql );
```

Add the new Context Action to a Context Definition record. In this instance, it will be added to the definition for the incidents table and the affected.item field. Click on the next available line under Action Name and choose fill. You will be presented with a QBE of available Context Actions. Double click on "Open Changes for this Service" and add an Action Condition of true. Save the record.

Context Definition

Table:

Field:

State:

Profile:


Action Name	Action Condition
Open Incidents for this Service	true
Closed Incidents for this Service	true
Possible workarounds for this Service	true
Open Changes for this Service	true

Ensure that there is a scmessage record for the action defined in the contextDefinition record above. Within Database Manager search on the scmessage table for a class of contextAction. If there is no scmessage record for the newly created action, add a record with the Action Name from the Context Definition record above in both the Message Number and Text fields.

Creating a new context wizard

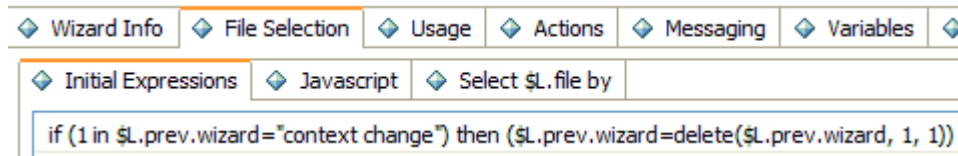
Since the new context wizard will function much like the existing wizards the most efficient method will be to modify the necessary fields in, for example, context incident and choose to “Add” the new wizard.

From the System Navigator choose Tailoring > Wizards. Enter context for the Wizard Name and Search. Select context incident and change the Wizard Name from context incident to context change.



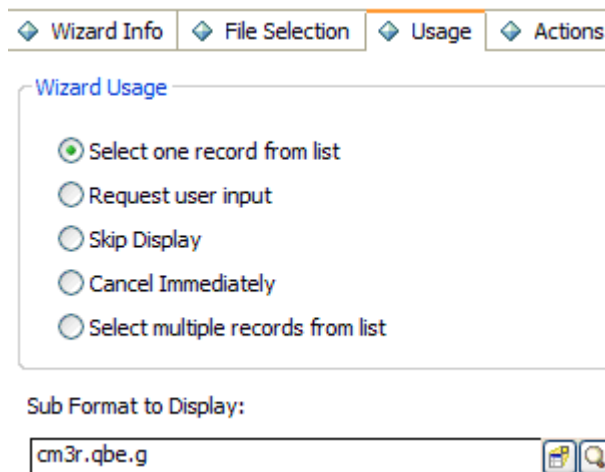
Wizard Name: context change

On the File Selection tab Initial Expressions change the value assigned to `$.prev.wizard` to “context change”



```
if (1 in $.prev.wizard="context change") then ($.prev.wizard=delete($.prev.wizard, 1, 1))
```

On the Usage tab change the Sub Format to Display to a Change Management QBE format such as `cm3r.qbe.g`.



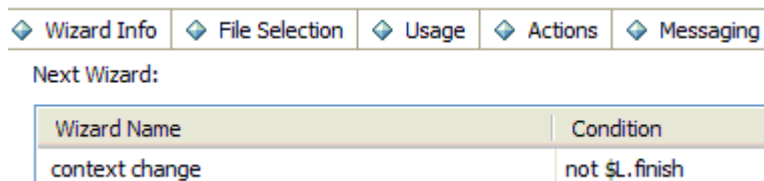
Wizard Usage

- Select one record from list
- Request user input
- Skip Display
- Cancel Immediately
- Select multiple records from list

Sub Format to Display:

cm3r.qbe.g

On the Next Wizard tab change the Wizard Name to context change



Wizard Name	Condition
context change	not \$.finish

Choose Add to save the changes as a new wizard. The final step is to add this new wizard to the context choose wizard to ensure that it will be run. On the Next Wizard tab add context change under Wizard Name. The Condition will be the same as the other wizards with the exception that the value assigned to `$G.context.table` = “cm3r”. Save the changes to the wizard.

Next Wizard:

Wizard Name	Condition
context incident	<code>\$.finish~=true and \$.selected.action.index in \$G.context.table="probsummary"</code>
context interaction	<code>\$.finish~=true and \$.selected.action.index in \$G.context.table="incidents"</code>
context knownerror	<code>\$.finish~=true and \$.selected.action.index in \$G.context.table="knownerror"</code>
context change	<code>\$.finish~=true and \$.selected.action.index in \$G.context.table="cm3r"</code>

For more information

Please visit the HP Software support Web site at:

www.hp.com/go/hpsupport

This Web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued customer, you can benefit by being able to:

- Search for knowledge documents of interest
- Submit and track progress on support cases
- Submit enhancement requests online
- Download software patches
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Note: Most of the support areas require that you register as an HP Passport user and sign in. Many also require an active support contract.

To find more information about support access levels, go to the following URL:

www.hp.com/go/hpsupport/new_access_levels

To register for an HP Passport ID, go to the following URL:

www.hp.com/go/hpsupport/passport-registration

Technology for better business outcomes

© Copyright 2009 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

Linux is a U.S. registered trademark of Linus Torvalds. Microsoft and Windows are U.S. registered trademarks of Microsoft Corporation. UNIX is a registered trademark of The Open Group. JavaScript is a registered trademark of Sun Microsystems, Inc. in the United States and other countries. Oracle is a registered trademark of Oracle Corporation and/or its affiliates

