An OASIS DITA Adoption Technical Committee Publication

DITA 1.3 Feature Article: Using DITA 1.3 Troubleshooting

Author: Bob Thomas
On behalf of the DITA Adoption Technical Committee

Date: 12 May 2014

Table of Contents

OASIS (Organization for the Advancement of Structured Information Standards) is a not-for-profit, international consortium that drives the development, convergence, and adoption of e-business standards. Members themselves set the OASIS technical agenda, using a lightweight, open process expressly designed to promote industry consensus and unite disparate efforts. The consortium produces open standards for Web services, security, e-business, and standardization efforts in the public sector and for application-specific markets. OASIS was founded in 1993. More information can be found on the OASIS website at http://www.oasis-open.org.

The OASIS DITA Adoption Technical Committee members collaborate to provide expertise and resources to educate the marketplace on the value of the DITA OASIS standard. By raising awareness of the benefits offered by DITA, the DITA Adoption Technical Committee expects the demand for, and availability of, DITA conforming products and services to increase, resulting in a greater choice of tools and platforms and an expanded DITA community of users, suppliers, and consultants.

DISCLAIMER: All examples presented in this article were produced using one or more tools chosen at the author's discretion and in no way reflect endorsement of the tools by the OASIS DITA Adoption Technical Committee.

This white paper was produced and approved by the OASIS DITA Adoption Technical Committee as a Committee Draft. It has not been reviewed and/or approved by the OASIS membership at-large.

Document History

Revision	Date	Author	Summary
First Draft	2 February 2014	Thomas	Draft of the Committee Note -- Feature Article
Draft	22 May 2014	Thomas	Draft for committee vote

Using DITA 1.3 Troubleshooting

> *This article briefly describes the troubleshooting information type plus the DITA 1.3 features that support it. Several examples follow that show how to use the new troubleshooting features.*

Troubleshooting information

> *This topic describes the troubleshooting information type.*

What is troubleshooting?

Troubleshooting is an information type that provides corrective actions for changing the state of a product or a system to a state that is more desirable.

Simple troubleshooting

In its simplest form, troubleshooting information provides corrective actions that follows this pattern:

1. A condition or symptom. Usually the condition or symptom is an undesirable state in a system, a product, or a service that a reader may wish to correct.
2. A cause for the condition or symptom
3. A remedy for the condition or symptom that restores the system, product, or service to its normal state.

Complex troubleshooting

In complex cases, there may be more than one possible cause for a condition or a symptom. When this happens, each cause can be presented along with its associated remedy. These cause-remedy pairs serve as successive fixes users can try to eliminate an undesirable condition.

Embedded troubeshooting

Embedded troubleshooting information appears within tasks or descriptions. It is brief. Often, the condition or cause is implied by the information that surrounds it. Remedies can usually be conveyed with a single sentence.

Tip:

Extensive amounts of troubleshooting information ought never be embedded. Use troubleshooting topics instead.

Other corrective action information

Other corrective action information that follows the troubleshooting information type pattern are:

Alarm clearing	When something goes wrong, a system returns an alarm from a predefined set of alarms.
Error resolution	When something goes wrong, a system returns an error code from a predefined set of error codes.
Event response	When a significant event occurs, a system returns an event from a predefined set of events. Some of these events, while not errors, are nonetheless undesirable states that may warrant a response.

DITA 1.3 troubleshooting features

This topic describes DITA 1.3 support for the troubleshooting information type.

What are the troubleshooting features?

DITA 1.3 introduced the following troubleshooting features:

Troubleshooting topic	A new “troubleshooting” topic type that models troubleshooting information-type semantics.
tasktroubleshooting	This section-like element is a place for specifying the corrective action to take when a task fails.
steptroubleshooting	This step sub-element is a place for specifying the corrective action to take when a step fails.
Troubleshooting note-type	The “trouble” note-type is available alongside other note-types such as “caution”, “note”, or “tip”. The “trouble” note-type contains incidental corrective action that pertains to its surrounding content.

Why DITA 1.3 troubleshooting matters

With the new DITA troubleshooting markup, writers can create tightly focused information that helps readers resolve specific problems. Writers can now easily follow consistent patterns for troubleshooting information. This consistency expresses itself in predictable organization, titles, and icon graphics that lets readers quickly locate troubleshooting information. Companys save money when their users resolve problems without contacting technical support.

How do you use it?

Learn more about how to use these features by reading through the examples that follow. Addtionally, an annotated template for the troubleshooting topic appears at the end of this article.

Troubleshooting-topic examples

Simple troubleshooting topic

This is a basic troubleshooting-topic application.

Scenario

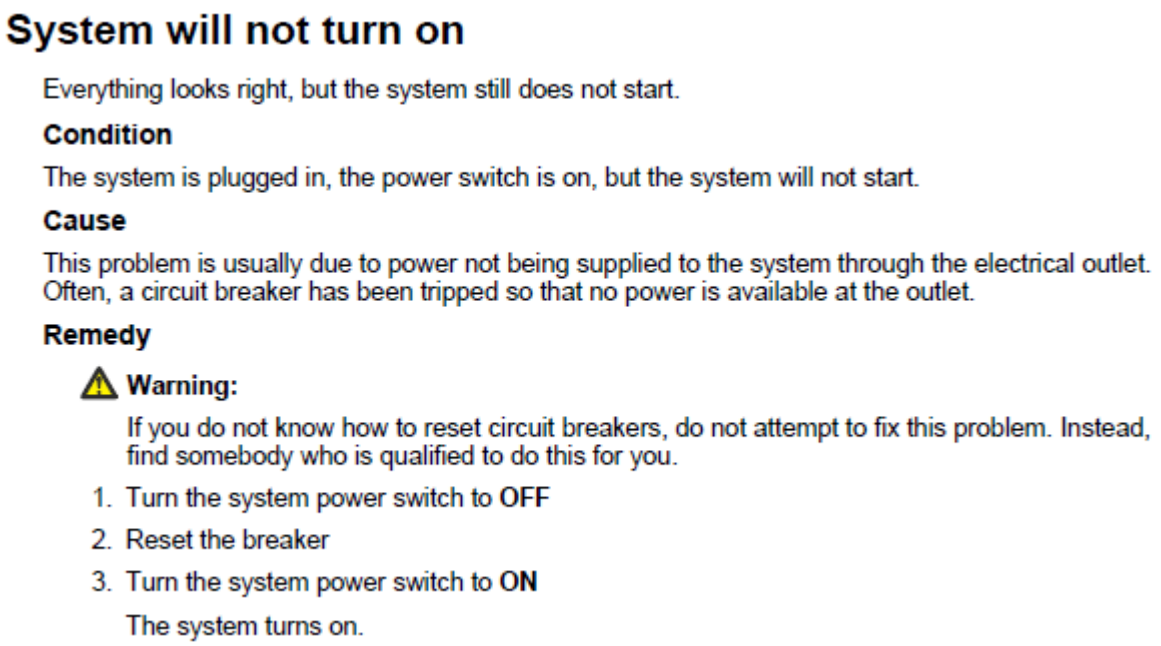
Name	Tripped circuit breaker
Description	The system is plugged in, the power switch is on, but the system will not start. This problem is external to the system, and it is almost always due to a tripped circuit breaker. The system is a low-power consumer product that runs on household electricity.

Discussion

We will use a troubleshooting topic with a single troubleSolution element to document the corrective action for this problem. In the tag view, pay close attention to the XML comments embedded within the topic markup. These comments contain usage advice.

Output

Before looking at the tag view, look at the output first:



System will not turn on

Everything looks right, but the system still does not start.

Condition

The system is plugged in, the power switch is on, but the system will not start.

Cause

This problem is usually due to power not being supplied to the system through the electrical outlet. Often, a circuit breaker has been tripped so that no power is available at the outlet.

Remedy

Warning:

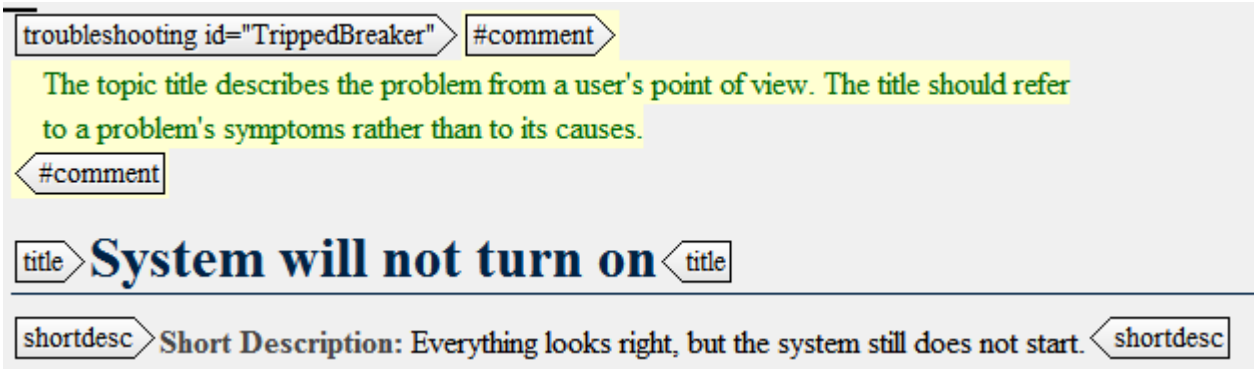
If you do not know how to reset circuit breakers, do not attempt to fix this problem. Instead, find somebody who is qualified to do this for you.

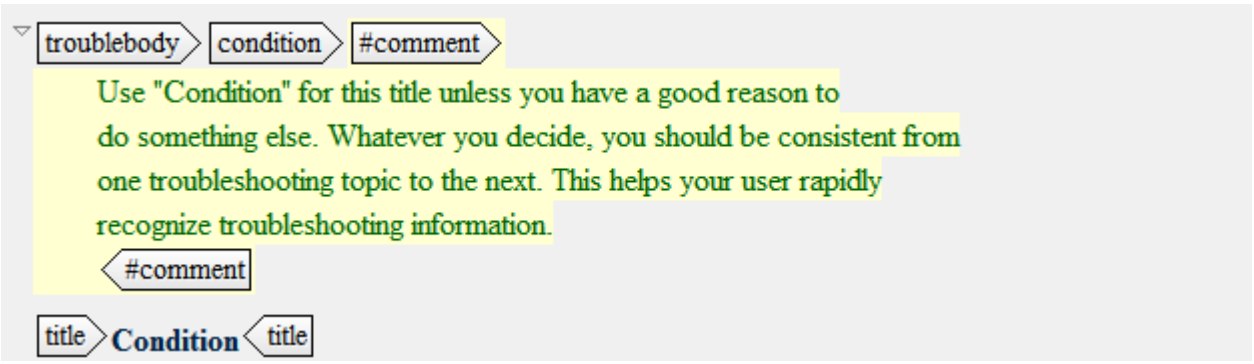
1. Turn the system power switch to **OFF**
2. Reset the breaker
3. Turn the system power switch to **ON**

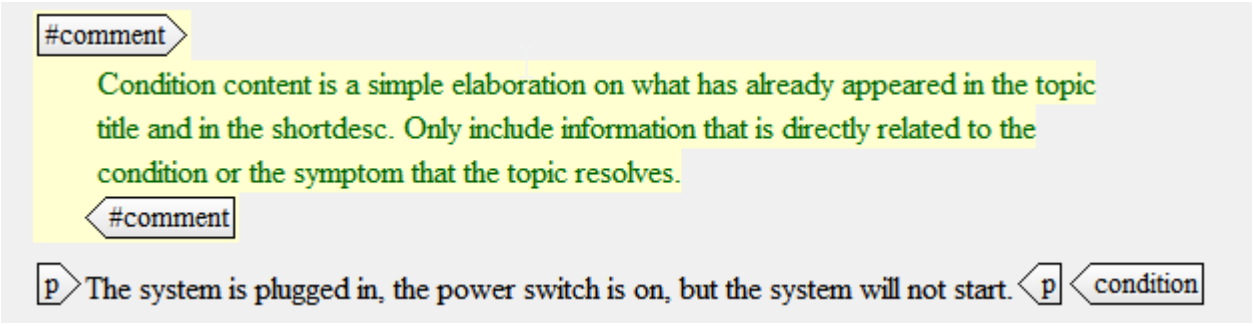
   The system turns on.

Tag view

Collectively, the following series of XML editor screen-shots form a complete troubleshooting topic. They have been split into several segments for formatting reasons.







#comment

Keep it simple with troubleSolution: one cause followed by one remedy. There may be cases where you have to do something different, but don't go there unless you absolutely must. #comment

troubleSolution cause #comment

Use "Cause" for this title unless you have a good reason to do something else. Consistency across topics is important. #comment

title **Cause** title

#comment

Cause content should only describe the problem origins that are fixed by the cause's companion remedy element. #comment

p This problem is usually due to power not being supplied to the system through the electrical outlet. Often, a circuit breaker has been tripped so that no power is available at the outlet. p

cause

remedy #comment

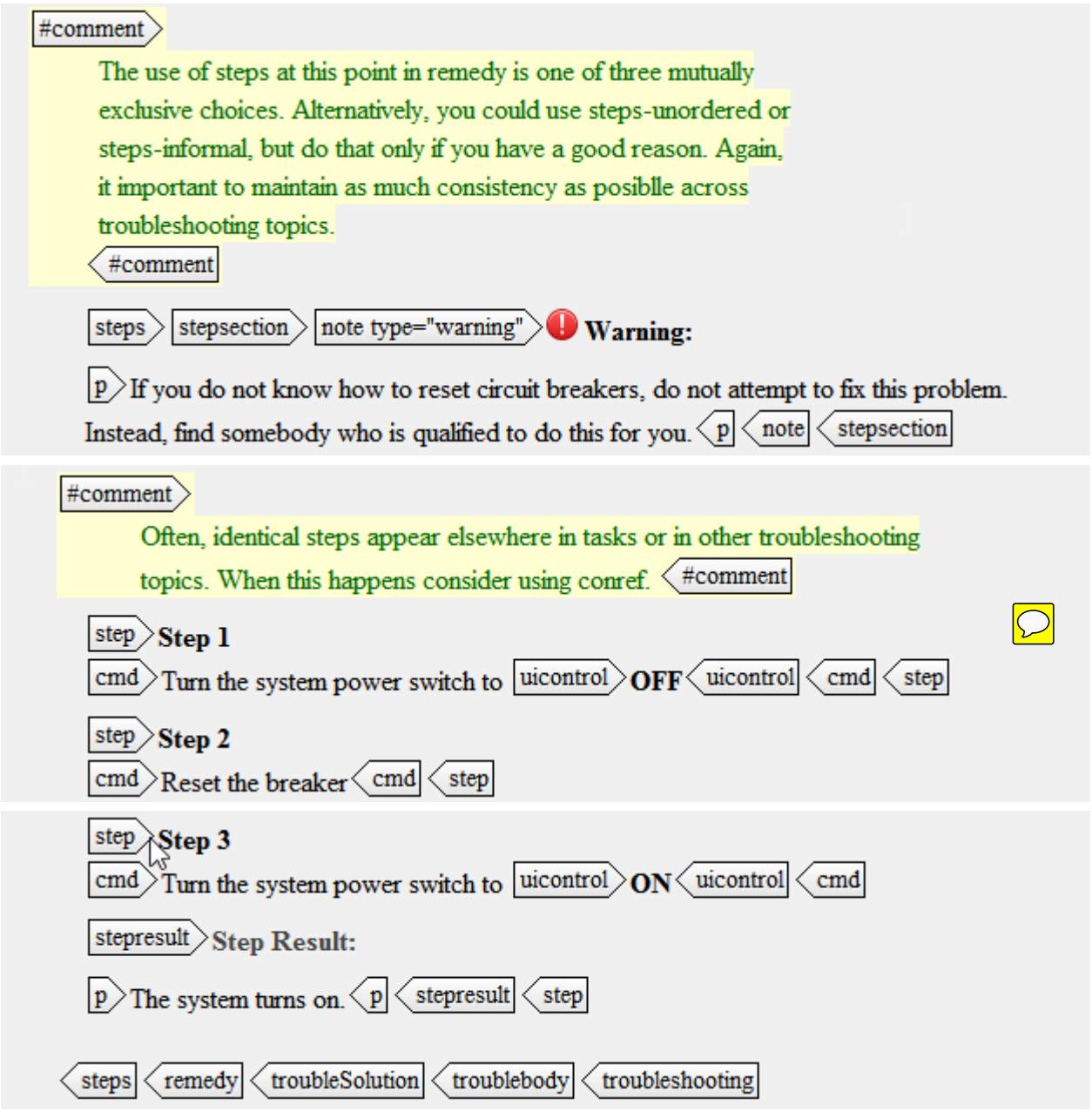
Use "Remdy" for this title unless you have a good reason to do something else. Consistency across topics is important. #comment

title **Remedy** title

#comment

The responsibleParty is the role of the person performing the remedy. This element is optional. Often, stylesheets will use this content as metadata and not output it directly. Do not use this element unless your stylesheets support it and you have a specific purpose in mind for using it. #comment

responsibleParty electrician responsibleParty



Multiple solutions troubleshooting topic

This is a complex troubleshooting topic application.

Scenario

Name	Cannot log in to the system.
Description	A customer cannot log in to the system. The reasons for this could be: no account exists, the user forgot their user id, or the user forgot their password. The user needs to contact support if none of the remedies resolve this problem.

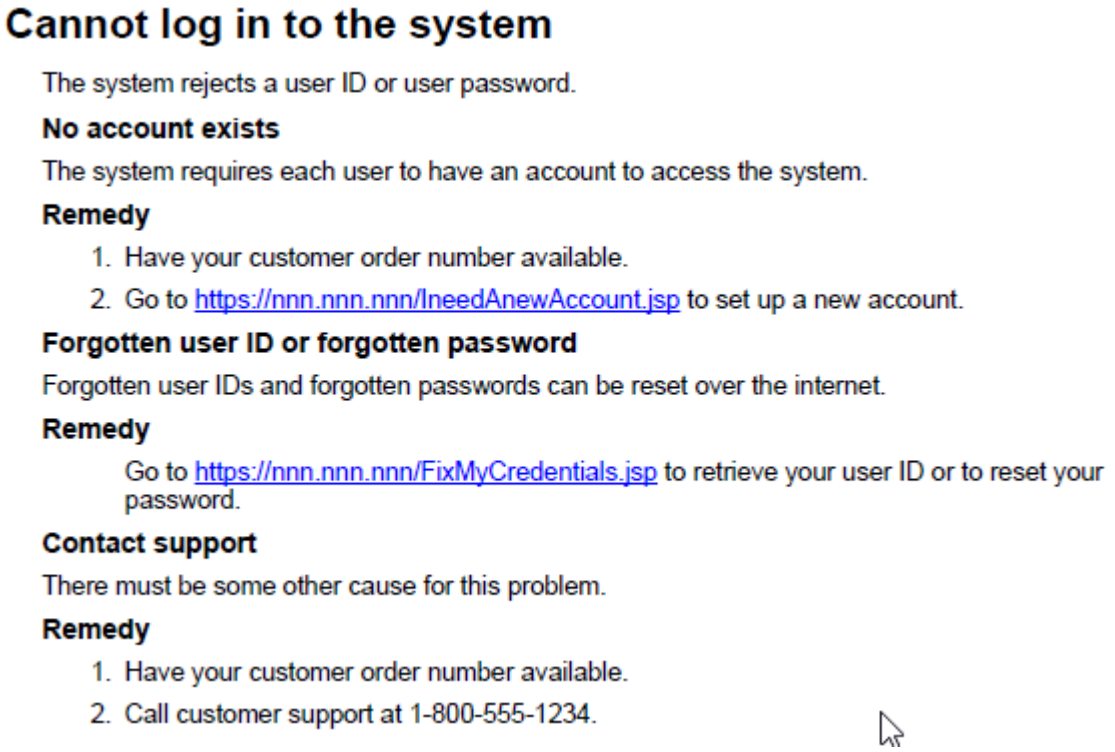
Discussion

We will use a troubleshooting topic with three troubleSolution elements to present a series of potential fixes (fallbacks) for this problem. The idea is that the user will try each troubleSolution until the problem is resolved. The first title in each troubleSolution is in the cause element, and it briefly describes the cause instead of using a consistent title such as "Cause." In multiple troubleSolution scenarios, descriptive labeling helps users diagnose problems quicker. Therefore, it is more important than using a consistent title.

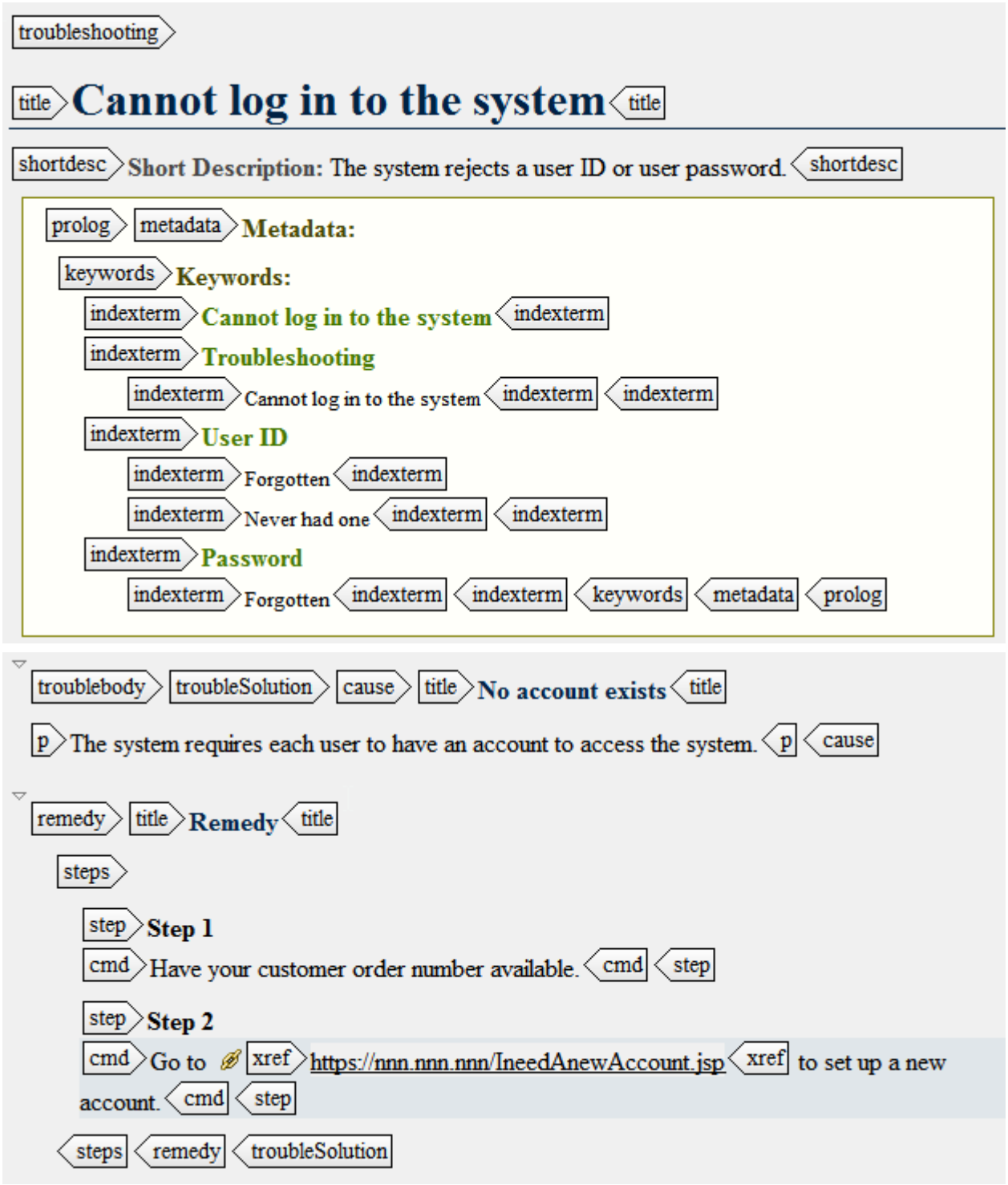
No condition element is used here. That is because all pertinent information about the condition has already been given in the topic title and in the shortdesc.

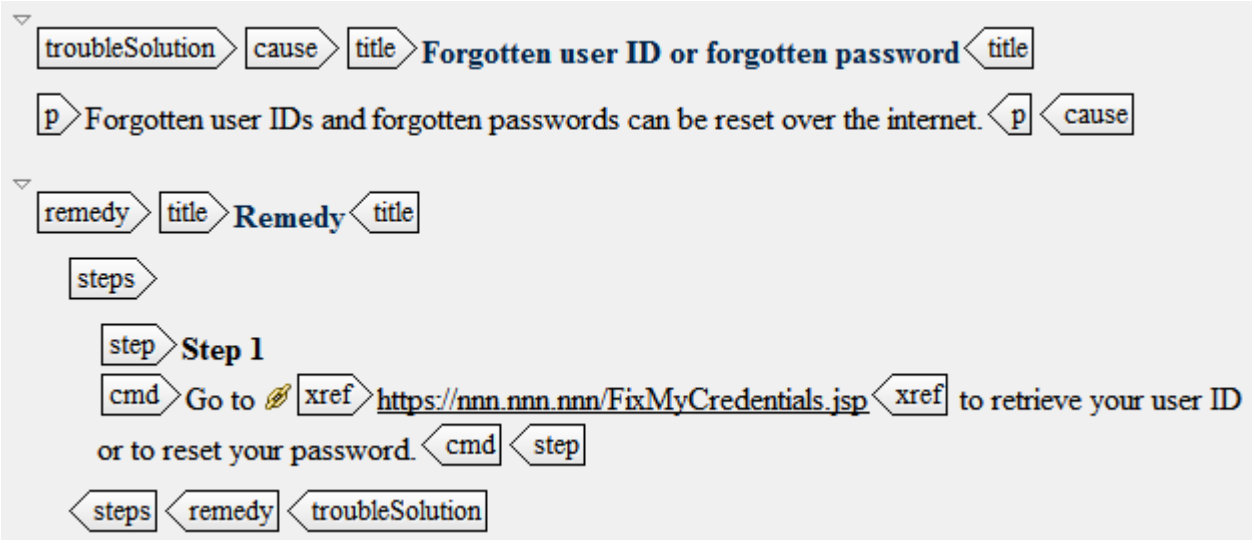
The first troubleSolution contains the "no account exists" cause/remedy. This troubleSolution must be first because there is no point in a user trying to reset account credentials for a non-existing account. The second troubleSolution contains the "forgotten user id/forgotten password" cause/remedy. The third troubleSolution contains the final fallback: calling customer support. In practical applications, this third troubleSolution would be reused through conref.

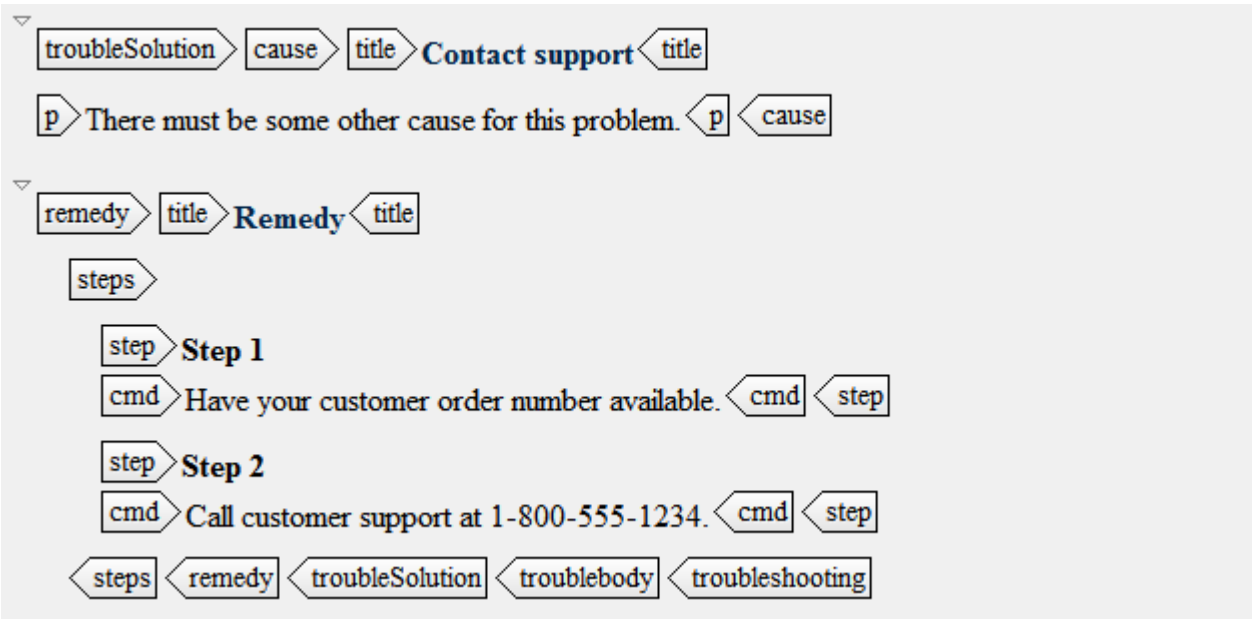
Output



Tag view







Alarm troubleshooting topic

This shows how troubleshooting-topic can contain alarm information.

Scenario

Name Alarm S503

Description The system issues alarm S503 whenever its connection with the network controller has been unavailable for more than 2 minutes. This is an intermediate-level alarm. The impact to the system is moderate. Circuit card TN2037 provides the system's LAN connectivity.

Possible causes for this problem are: 1) TN2037 is in the off-line state, 2) the wiring between the TN2037 and the network controller may have disrupted, 3) the network controller is experiencing system problems, or 4) the TN2037 is defective.

Discussion

We will use a troubleshooting topic with four troubleSolution elements to document this alarm along with its associated corrective actions. Note the use of defintion list within the condition element to organize the alarm's classification information.

Output

S503 - communication link failure

The communication link between the system and the network controller has been out of service for more than two minutes.

Condition

Alarm code	S503
Level	Intermediate
Impact	Moderate

The system issues alarm S503 whenever its connection with the network controller has been unavailable for more than 2 minutes. Circuit car TN2037 provides the system's LAN connectivity.

TN2037 is off-line

The TN2037 provides system connectivity to the LAN which is connected to the network controller. If the TN2037 is in the “off-line” state, no comunication can happen between the system and the network controller.

Remedy

1. Log in to the system adminstration interface.
2. Go to **Tools** > **Administration** > **Network**.
3. Select **Restart the network interface**

Network controller is having problems

There may be problems at the network-controller end of the link.

Remedy

Contact the network-controller administrator to find out whether there are problems with the nework controller.

LAN wiring is defective

The electrical connection between the system and the network controller may be defective.

Remedy

1. Perform a Network link integrity test.

   If the netowrk link passes its test, quit these remedy steps and proceed to Defective TN2037.
2. If the link is defective, replace the LAN cable between the TN2037 and the Ethernet hub.

Defective TN2037

Circuit card TN2037 is defective, and it must be replaced.

Remedy

Elided for this example. In real output this would be the first of several steps.

Tag view

troubleshooting id="AlarmTroubleshootingTopicSource"

title **S503 - communication link failure** title

shortdesc **Short Description:** The communication link between the system and the network controller has been out of service for more than two minutes. shortdesc

troublebody condition title **Condition** title

dl dlentry dt **Alarm code** dt

dd p S503 p dd dlentry

dlentry dt **Level** dt

dd p Intermediate p dd dlentry

dlentry dt **Impact** dt

dd p Moderate p dd dlentry dl

p The system issues alarm S503 whenever its connection with the network controller has been unavailable for more than 2 minutes. Circuit car TN2037 provides the system's LAN connectivity. p condition

troubleSolution cause title **TN2037 is off-line** title

p The TN2037 provides system connectivity to the LAN which is connected to the network controller. If the TN2037 is in the " q off-line q " state, no comunication can happen between the system and the network controller. p cause

remedy title **Remedy** title

responsibleParty administrator responsibleParty

steps

step **Step 1**

cmd Log in to the system adminstration interface. cmd step

step **Step 2**

cmd Go to menucascade uicontrol **Tools** uicontrol > uicontrol **Administration** uicontrol > uicontrol **Network** uicontrol menucascade . cmd step

step **Step 3**

cmd Select uicontrol **Restart the network interface** uicontrol cmd step

steps remedy troubleSolution

troubleSolution cause title **Network controller is having problems** title

p There may be problems at the network-controller end of the link. p cause

remedy title **Remedy** title

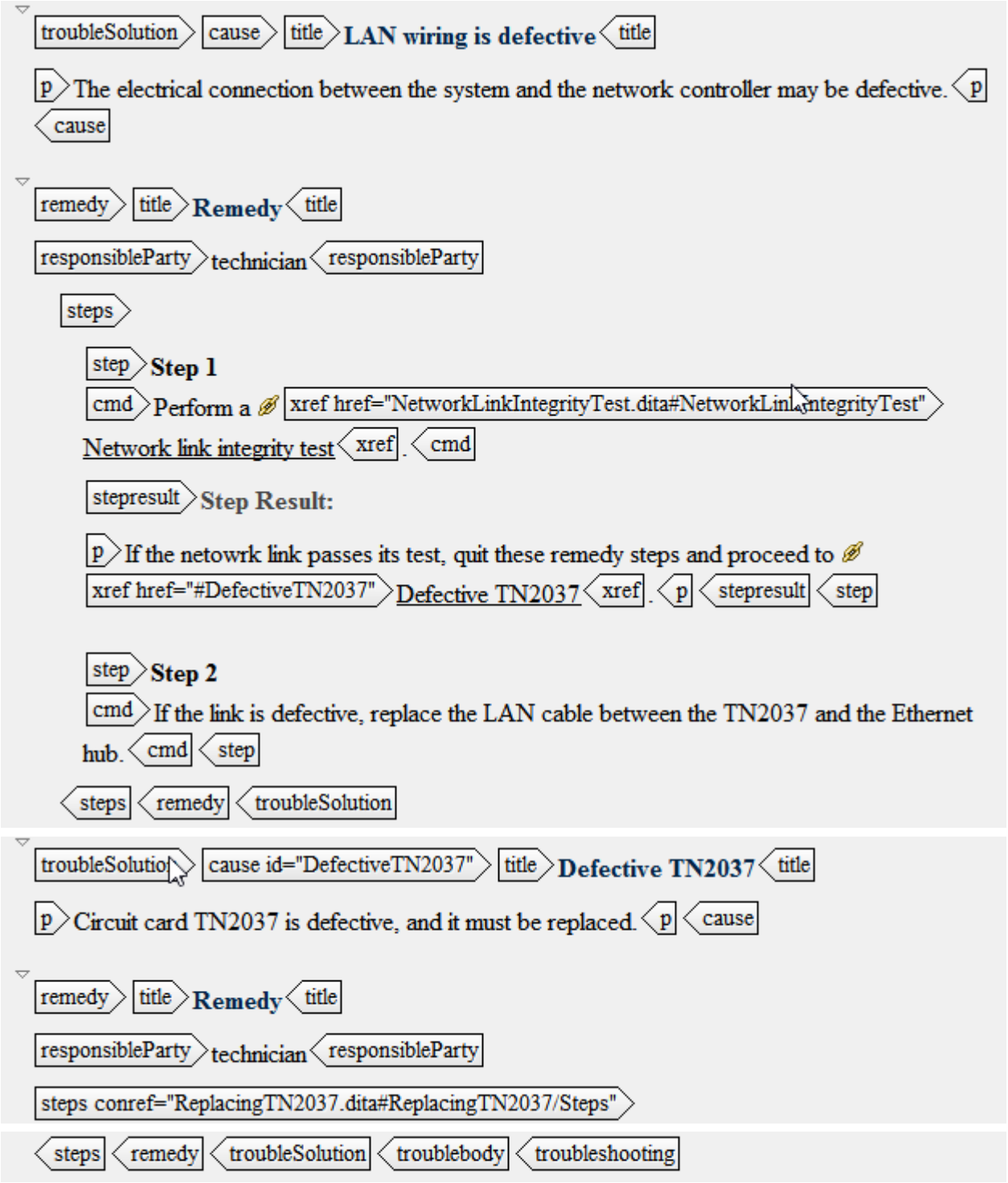
responsibleParty administrator responsibleParty

steps

step **Step 1**

cmd Contact the network-controller administrator to find out whether there are problems with the nework controller. cmd step

steps remedy troubleSolution



Embeded troubleshooting examples

Task-releated

This section contains examples showing streptroubleshooting and tasktroubleshooting

Discussion

The steptroubleshooting and tasktroubleshooting elements were designed to give users immediate, brief information about how to resolve problems when an undesirable result happens. This means writers should limit content to discussing condition, cause, and then perhaps a single-step. If lists containing instructions are present in the content, that is too much. The writer should re-design the content so that the instructions are in a separate task topic or are part of a troubleshooting topic. The writer should then refer the user to that other topic.

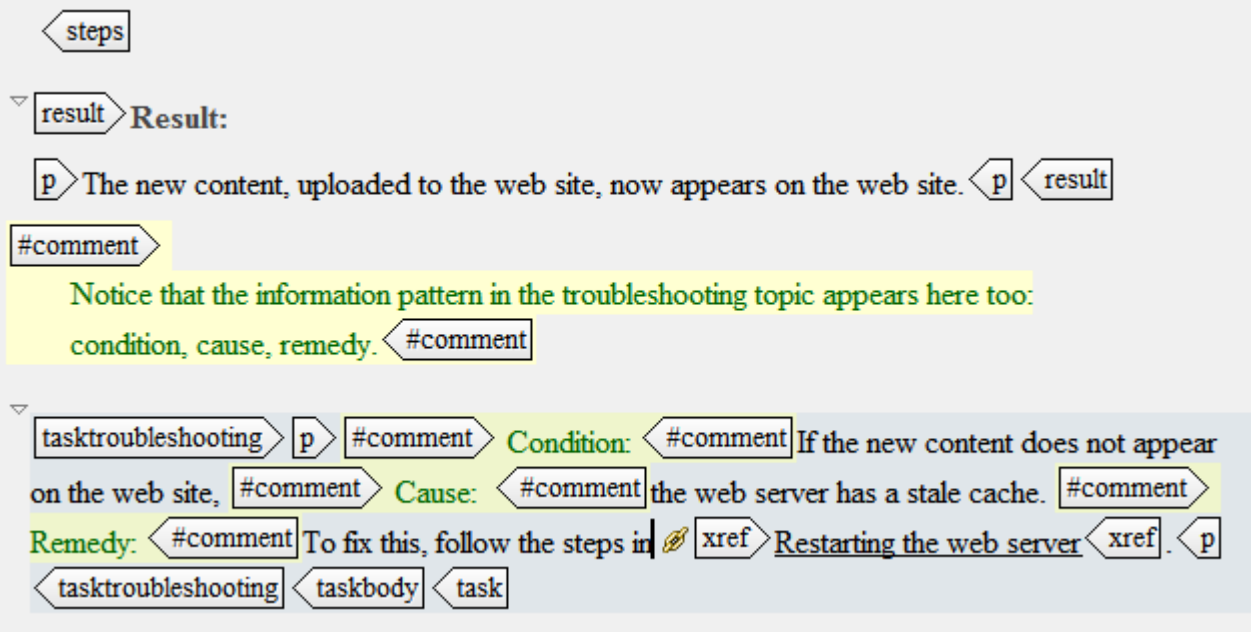
Task troubleshooting example

Scenario

Name Web site update fails

Description A customer follows the steps in a task for updating web site content, but the web site is not showing the new content. This could be due to a stale web server cache. The customser should restart the web server.

Tag view



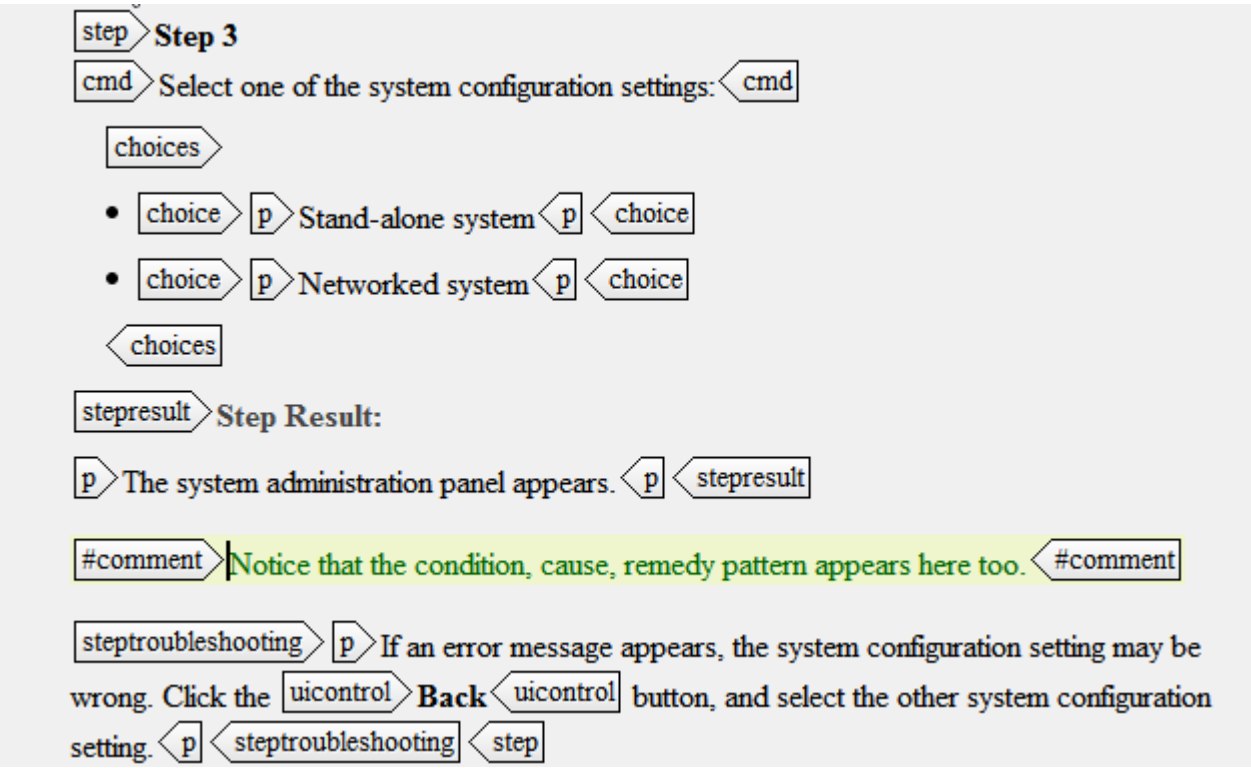
Step troubleshooting example

Scenario

Name Wrong option in step

Description A user has to select one of two system configuration settings in a step. If they select the wrong one, an error message displays. To recover, they need to click the **Back** button and select the other configuration setting.

Tag view

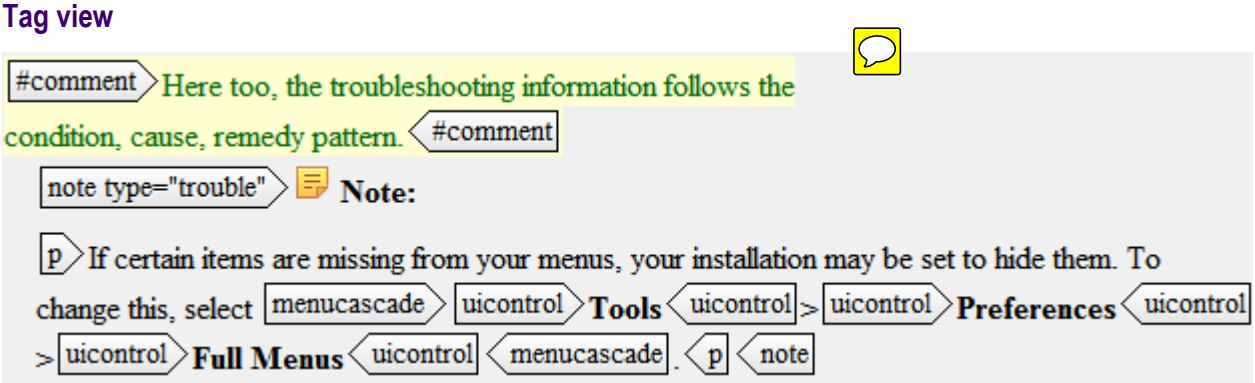


Troubleshooting note-type

Scenario

Name	Cannot see all menu items
Description	By default, a software product hides certain advanced menu items, however the product's documentation describes all of menus items. Customers following the documentation along on their systems have become confused when they did not see all of the menu items that were being described. The fix for this is to have the customer change their preferences to "Show Full Menus".

Tag view

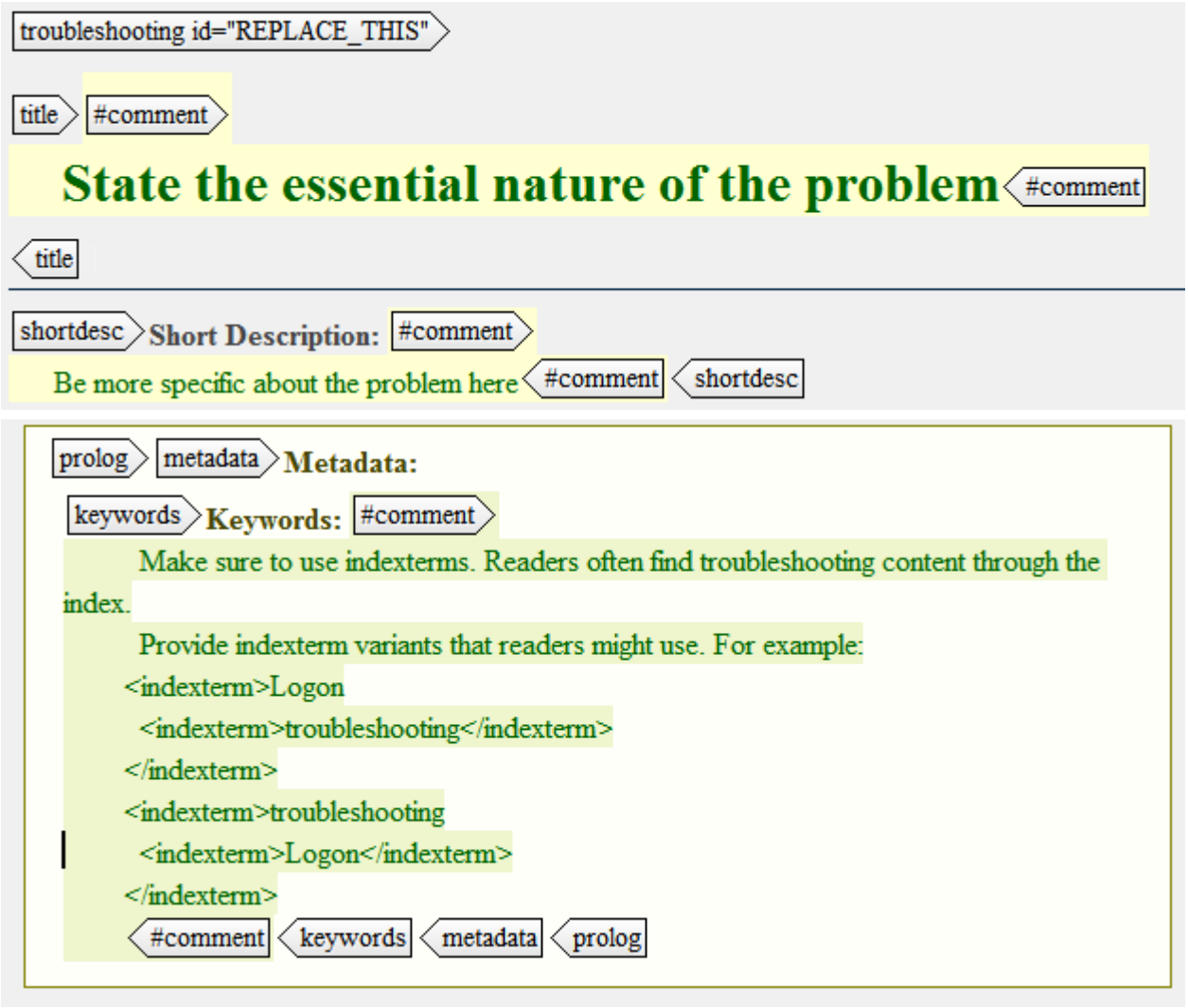


Annotated troubleshooting template

This is a DITA troubleshooting topic template with usage comments.

XML comments describe each element and how to use it. Formal descriptions for each troubleshooting element can be found in the *Darwin Information Typing Architecture (DITA) Version 1.3 OASIS Standard* Language Specification section.

Tag view



troublebody condition #comment

The topic title and the shortdesc should have already told your reader a lot about the condition that this topic seeks to fix. Use condition to expand upon that. Do not simply repeat what you have already put into the title and the shortdesc. Condition is the appropriate place to put impact and severity information. If multiple solutions exist and their relationships with each other are complex, you can discuss that here. #comment

title #comment

Optional title. Use "Condition". #comment title

p **p** p condition

troubleSolution #comment

troubleSolution is meant to hold pairs of cause and remedy. Occassionally, you might have cause without remedy or remedy without cause, but that should be rare. Be sure to order mutliple troubleSolution elements in a sequence that makes sense. For example, order them by the liklihood of a cause occuring. You may wish to deviate from that if a remedy for a less likely cause is much easier to try. Remember to use conref for troubleSolution elements that are the same across multiple troubleshooting topics. #comment

cause title #comment

Optional title. Use "Cause". For topics with mutliple troubleSolutions, state the essential nature of this particular cause instead of just using "Cause". #comment title

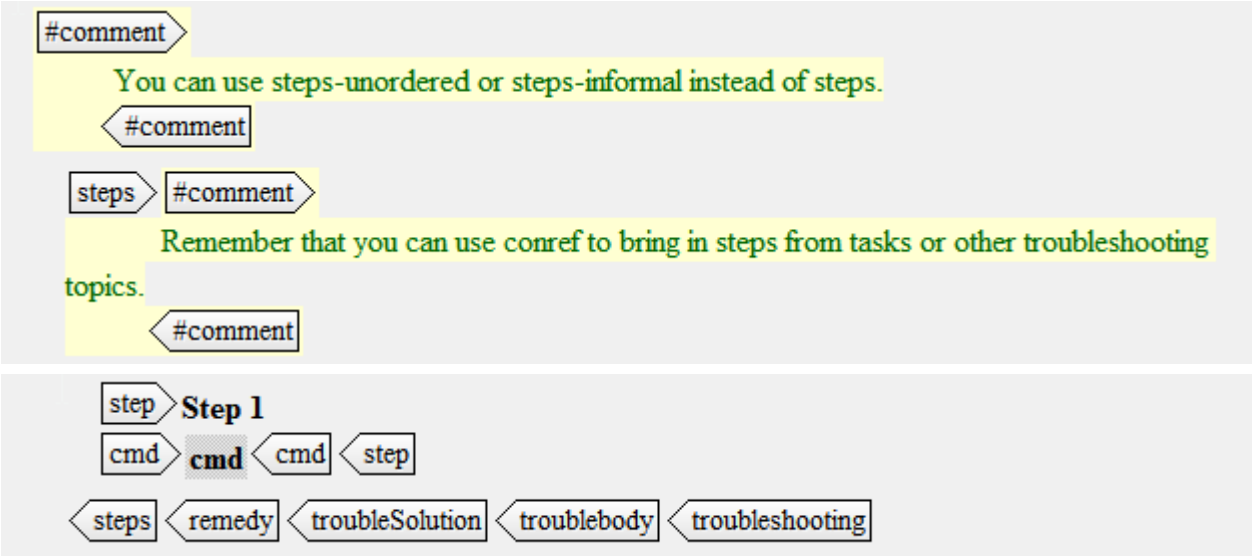
p **p** p cause

remedy title #comment

Optional title. Use "Remedy" or "Solution". #comment title

responsibleParty #comment

Optional. Use this element to indicate the role of who ought to be performing the steps in the remedy. Here are some examples: "engineer", "customer", "field-support". #comment responsibleParty



XML view

Here is the template's XML. It can be copied and pasted into an XML editor.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE troubleshooting
  PUBLIC "-//OASIS//DTD DITA Troubleshooting//EN"
  "troubleshooting.dtd">
<troubleshooting id="REPLACE_THIS">
  <title><!-- State the essential nature of the problem --></title>
  <shortdesc><!-- Be more specific about the problem here --></shortdesc>
  <prolog>
    <metadata>
      <keywords><!--
          Make sure to use indexterms. Readers often find troubleshooting content through the index.
          Provide indexterm variants that readers might use. For example:
        <indexterm>Logon
          <indexterm>troubleshooting</indexterm>
        </indexterm>
        <indexterm>troubleshooting
          <indexterm>Logon</indexterm>
        </indexterm>
        -->
      </keywords>
    </metadata>
  </prolog>
  <troublebody>
    <condition><!--
        The topic title and the shortdesc should have already told your reader a lot about the condition
        that this topic seeks to fix. Use condition to expand upon that. Do not simply repeat what you have
        already put into the title and the shortdesc. Condition is the appropriate place to put impact
        and severity information. If multiple solutions exist and their relationships with each other are
        complex, you can discuss that here.-->
      <title><!-- Optional title. Use "Condition". --></title>
      <p></p>
    </condition>
    <troubleSolution><!--
        troubleSolution is meant to hold pairs of cause and remedy. Occassionally, you might have cause
        without remedy or remedy without cause, but that should be rare. Be sure to order mutliple
        troubleSolution elements in a sequence that makes sense. For example, order them by the liklihood
        of a cause occuring. You may wish to deviate from that if a remedy for a less likely cause is much
        easier to try. Remember to use conref for troubleSolution elements that are the same across
        multiple troubleshooting topics.-->
      <cause>
        <title><!--
          Optional title. Use "Cause". For topics with mutliple troubleSolutions, state the essential
          nature of this particular cause instead of just using "Cause".-->
        </title>
        <p></p>
      </cause>
```

```
      <remedy>
        <title><!-- Optional title. Use "Remedy" or "Solution". -->
        </title>
        <responsibleParty><!--
          Optional. Use this element to indicate the role of who ought to be performing the steps in the
          remedy. Here are some examples: "engineer", "customer", "field-support".
        -->
        </responsibleParty>
        <!-- You can use steps-unordered or steps-informal instead of steps. -->
        <steps>
          <!-- Remember that you can use conref to bring in steps from tasks or other troubleshooting topics. -->
          <step>
            <cmd/>
          </step>
        </steps>
      </remedy>
    </troubleSolution>
  </troublebody>
</troubleshooting>
```