

# **Whitepaper: DITA Badges**

# Contents

- Content Badging in OASIS DITA (FIRST REVIEW).....3**
  - A typical use case for content badging.....4
  - Badge design and implementation..... 5
  - Establishing logic.....6
  - Badging and scope.....7
  - Going on a test drive.....7
  - Placing badges in your topics..... 8
    - Topic placement.....8
    - Section placement.....8
  - Referencing badges in your topics.....8
  - Referencing badges in shared topics.....9
  - Conclusion..... 11

# Content Badging in OASIS DITA (FIRST REVIEW)

OASIS DITA Adoption whitepaper on content badging.

All sample badges and supporting DITA examples are available to clone at <https://github.com/StanDoherty/dita-badges> (need OASIS home).

Badging is a popular topic on several fronts. The three most popular forms of badging are:

- *Achievement badging*: Organizations set up collections of social media "badges" to recognize the involvement and achievement in employees, partners, and customers. Imagine if someone developed a technical certification program for DITA and awarded badges to program participants who passed tests for particular features.



- *Status badging*: Collaborative development platforms allow administrators to incorporate macros in their portal pages that display the current status of builds, workflow stages, contributions, and test coverage.

tests 469 passed, 10 failed

build passing

build unknown

build failing

build failure

- *Content badging*: Content development organizations often decide to have one publication document multiple, closely-related products. Content badges alert readers whether the relevance of a particular topic, section, or element is restricted to a specific product or release version. Here are some samples of icon-based and tag-based badges.



This topic does not apply to Cloud Compute.



This topic applies exclusively to Cloud Compute.

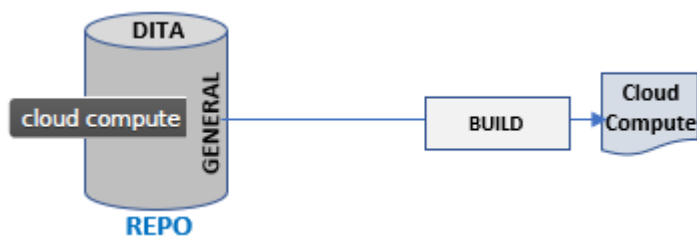
cloud compute not supported This section does not apply to Cloud Compute.

cloud compute supported This topic applies exclusively to Cloud Compute.

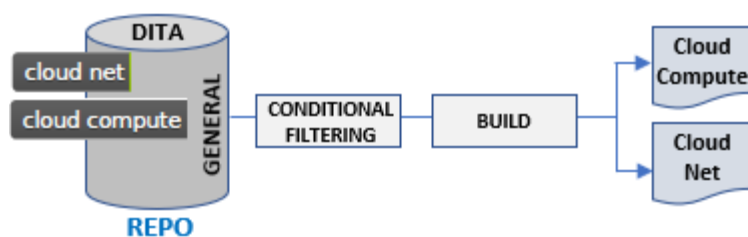
Content badging is a medium- to high-risk content strategy for compound publications, that is, publications that document more than one product or version.

## A typical use case for content badging

A startup named CloudSquared develops resource monitoring software for cloud-based frameworks. For its first product, Cloud Compute Monitoring, the content development team develops a complete documentation set.

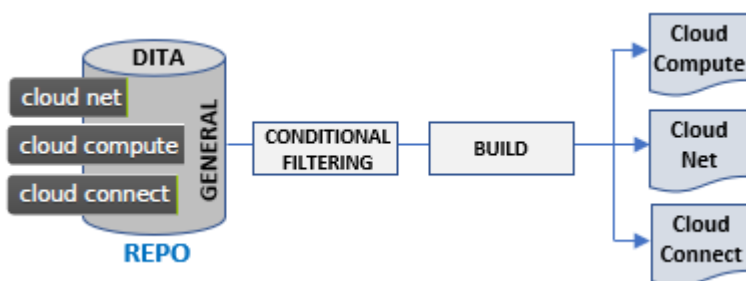


A year later CloudSquared releases Cloud Net Monitoring so the content development team uses DITA conditional filtering to publish separate Cloud Compute and Cloud Net publications.



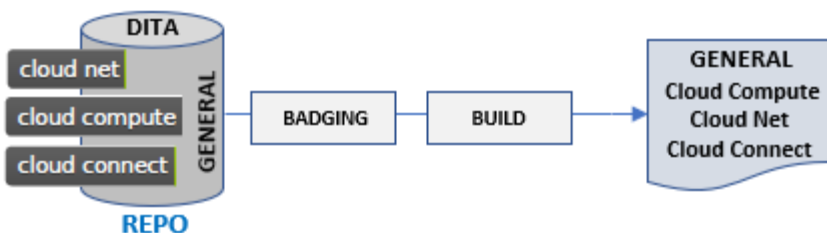
Content developers assign filtering attributes such as `@product="cloud-compute"` or `@product="cloud-net"` to topics, sections, or elements that need to be filtered OUT of a product-specific deliverable.

The following year, CloudSquared releases Cloud Connect and completes the monitoring suite. The writing team then uses conditional filtering to publish three separate publications.



Content developers add the filtering attribute `@product="cloud-connect"` to those topics, sections, or elements that need to be filtered OUT of the other two product-specific deliverables.

Six months after this launch, Marketing receives feedback that emphasizing common design and services across the three suite components will enhance customer perception that CloudSquared can consolidate its achievements and move on to new product lines. Toward that end, Product Management requests that the content development team consolidate the three, product-specific publications into one, multi-product publication. All generic and product-specific information would need to appear in the same deliverable.



When transitioning from conditional filtering to badging, note that filtering metadata is *useful* in identifying where product-specific information lives in your sources but *useless* as markup to implement badging. In DITA, the logic to include or exclude elements tagged with filtering attributes such as `@product` or `@audience` lives in DITAVAL scripts outside the DITA topics themselves.

Filtering markup in a topic:

```
<p @product="cloud-connect">Cloud Connect is great.</p>
```

Filtering markup in a DITAVAL script:

```
<val>
  <prop att="product" val="cloud-connect" action="exclude" />
</val>
```

Badging markup:

```
<p id="p_badge_icon_cloud-connect_section_yes">
  <image href="images/badge_cloud-connect_small_yes.png"></image>
  <ph>This section applies exclusively to Cloud Connect.</ph>
</p>
<p>Cloud Connect is great.</p>
```



Filtering does a brilliant job supporting a many-to-one relationship between source content and delivered content. Many product-specific variations are filtered out to produce that one product-specific deliverable. Badging involves a many-to-many relationship. All the product-specific variations in the sources need to be accounted for in a multi-product deliverable.

The real challenges with badging involve logic.

## Badge design and implementation

Logic and planning problems of this sort are best approached visually. Design exercises such as card sorting or affinitytization depend upon multiple team members seeing all the variables and iteratively testing various arrangements or groupings. Similarly, making a first, complete pass at designing all the content badges that you might need will accelerate the process of developing and refining the logic of your badging. If you have time to code the badges in a DITA library, that's great -- but sticky notes or 3x5 cards will suffice for team discussions early on. Move the cards around. Tape them to whiteboards. Staple them to printouts of your existing docs. Any exercise that simulates you applying a badge to your content and then asking whether that makes sense is goodness.

Regardless of how you implement badging in DITA, content badges have four basic components:

Visual cue	Content ID	Content scope	Content switch
	Cloud Net	This section	applies to . . .
	Cloud Net	This topic	does not apply to . . .

Whether you and your graphic designers choose icons or tags, building a library of reusable DITA badges is straightforward. Consider starting with just one library for badging, call it `library_content-badges.dita`.

The following markup for a positive badge . . .

```
<p id="p_badge_icon_cloud-connect_section_yes">
  <image href="images/badge_cloud-connect_small_yes.png">
    <alt>This section applies exclusively to Cloud Connect.</alt>
```

```

</image>
<ph> This section applies exclusively to Cloud Connect.</ph>
</p>

```

... generates this badge in output.



**This section applies exclusively to Cloud Connect.**

The following markup for a negative badge ...

```

<p id="p_badge_tag_cloud-compute_topic_no">
  <image href="images/tag_cloud-compute_not-supported.svg">
    <alt>This topic does not apply to Cloud Compute.</alt>
  </image>
  <ph> This topic does not apply to Cloud Compute.</ph>
</p>

```

... generates this badge in output.

**cloud connect not supported** This topic does not apply to Cloud Connect.

Time that you spend organizing these badge definitions in a DITA library topic has a big payoff for content developers on your team. If writers can easily find the appropriate badge, they'll thank you -- eventually.

section

title

Sections: Icon-based badges

title

p

[ This section applies exclusively to Cloud Compute. ]

This section applies exclusively to Cloud Compute.

p

p

[ This section does not apply to Cloud Compute. ]

This section does not apply to Cloud Compute.

p

section

title

Topics: Tag-based badges

title

p

cloud compute supported

[ This topic applies exclusively to Cloud Compute. ]

This topic applies exclusively to Cloud Compute.

p

p

cloud compute not supported

[ This topic does not apply to Cloud Compute. ]


This topic does not apply to Cloud Compute.

p

## Establishing logic

If your badges are crisply defined and you apply them logically and consistently to your content, badging works. The first step toward providing logic for your publication is having a global statement about non-badged content.



Unless otherwise indicated with a product badge such as , all content applies equally to CloudSquared Cloud Compute, Cloud Net, and Cloud Connect.

This global statement sets the baseline logically. The only badges that the customer should expect to see would be ones that are exclusionary, identifying topics or sections that are *not* applicable to a particular product. The simpler your logic for inserting badges, the more consistently content developers can apply them and customers can understand them. In this scenario, you do not need to insert any inclusive badges, only exclusive ones. This is not unlike DITA filtering attributes that are designed, generally, to exclude content from processed output.

`cloud compute` **not supported** This topic does not apply to Cloud Compute.

`cloud net` **not supported** This section does not apply to Cloud Net.

`cloud connect` **not supported** This topic does not apply to Cloud Connect.

In our use case, you would then require only six badges -- 3 products x 2 scopes (topic or section).

## Badging and scope

---

When we content developers are looking at badged content in a DITA editor, we see where `<topic>` and `<section>` elements begin and end. We see the scope of the element relative to the topic displayed in our DITA editor. Unfortunately, these topic and section boundaries are neither visible nor recoverable to customers. In a running PDF, where does the current "topic" or "section" end? If we chunk multiple source topics so they generate one HTML5 page in output, wouldn't the topic boundaries be misleading? If I insert a topic-level badge to indicate that no content in the topic applies to a particular product, what happens if another writer conref's a section from my topic? That conref'd section has no section badge. Messy stuff.

You need to scope your topic-level or section-level badges to what you expect the customer will see on the page or in a browser, not what we see in our DITA sources. How do you figure that out? You test it out and show it to people in all your generated output formats.

Is it technically possible to create badges scoped to the level of DITA elements, for example paragraphs, figures, table rows, or phrases? Yes.

Practically, attempting to use the same badges for elements that you use for topics or sections can create a visual and logical train wreck. Teams that have successfully worked with badged documentation for multiple releases turn to `<note>` elements or in-line explanations for anything more granular than sections. Others use DITA flagging to color-code platform-specific information (accessibility - ouch).

Document the logic and the scoping of your badging strategy.

## Going on a test drive

---

Once you have normed on your logic and authoring guidelines, apply them to a small publication. Badging is messy stuff and you do not want to implement it for a complete doc set until you have received a "go-ahead" from your stakeholders. When they actually see a sample of your "badged" documentation, they may have objections or concerns that you had not factored into your initial design.

- *UXD*: "Shouldn't the graphic design of the badges conform to the new company UX guidelines?"
- *Support*: "Some customers are not careful readers and tend to ignore subtleties like these badges. They try something that is not appropriate for their product and then call us. Why are we doing this again? It is increasing our workload."
- *Test engineering*: "When we tested your procedures in the product-specific manuals, we could clearly identify issues with your writing. The new format is interesting, but we are never quite sure whether we are supposed to be testing all product procedures at once or be tiptoeing through the product-specific procedures one at a time."

- *Software engineering*: "Sometimes the differences between product features are not binary, not quite so simple as 'supported' or 'not supported'. Some features are "minimally supported" or "mostly supported."
- *Sales*: In pre-sales discussions, handing prospective customers manual that documents three products kinda confuses them if they are interested in buying one."
- *Marketing*: "I didn't realize that the docs would need so many badges. Can you make them smaller or put them in the margin somewhere?"
- *Legal*: "Our product documentation serves as the definitive product description. If the documentation does not accurately describe the product that the customer has purchased, we cannot recognize revenue."

Showing everyone a robust sample of badged documentation generates discussion across the organization about badging and other requirements for product documentation.

## Placing badges in your topics

---

Where you place badges is part common sense and part team preference.

### Topic placement

The following markup places the badge in the first sentence of the body of the topic.

```
<topic id="test1">
  <title>Topic title</title>
  <shortdesc>Short description</shortdesc>
  <body>
    <p>Topic-level badge reference</p>
  </body>
</topic>
```

This markup places the badge inside the `<abstract>` element at the beginning of the topic.

```
<topic id="test2">
  <title>Topic title</title>
  <abstract>
    <shortdesc>Short description</shortdesc>
    <p>Topic-level badge reference</p>
  </abstract>
  <body>
    <p>Running text</p>
  </body>
</topic>
```

### Section placement

The following markup places the badge in the first sentence of the `<section>`.

```
<section>
  <title>Section title</title>
  <p>Section-level badge reference</p>
  <p>Running text</p>
</section>
```

In this case, consistency is the important thing.

## Referencing badges in your topics

---

If you have built a library for your badges and assigned each of them a unique ID, you can use `@conref` or `@conkeyref` to insert them by reference them into your current topic.



Let's assume that you have created a badging library named `library_content-badges.dita` with a topic `@id` of `library1`. That library contains the following badge definition.

```
<p id="p_badge_tag_cloud-compute_topic_yes">
  <image href="images/tag_cloud-compute_supported.svg">
    <alt>This topic applies exclusively to Cloud Compute.</alt>
  </image>
  <ph> This topic applies exclusively to Cloud Compute.</ph>
</p>
```

To insert this badge using `@conref` (URL referencing), you would enter the following markup.

```
<p conref="library_content-badges.dita#library1/p_badge_tag_cloud-
connect_topic_yes"/>
```

To insert this badge using DITA keys, you must first define a key for the badging library in your current map.

```
<topicref keys="badgelib" href="library_content-badges.dita"/>
```

You can then reference the badge using `@conkeyref` (indirect referencing).

```
<p conkeyref="badgelib/p_badge_tag_cloud-connect_topic_yes"/>
```

The `@conref` and `@conkeyref` mechanisms reference the same badge definition in the same badge library file. Using indirect, key-based references is a wise investment because it will provide you with flexibility in managing multiple badge library files down the road. Change the key definition once, all `@conkeyref` references inherit the new badge definition automatically.

## Referencing badges in shared topics

What happens when the badged topics that support your three-product context get referenced by someone from another product group? At a minimum, they'll get a build error (missing key or reference). Ultimately their customers will not understand what is going on.

Be practical. Badging is a work-around and will not scale beyond a workgroup. Clone the shared topic, remove the badges from the clone, and move on.

If you *must* share a badged topic across teams but do not want the badges to appear in all contexts, consider one of the more underutilized reuse mechanisms in DITA, `conref` push.

By default, `@conrefs` and `@conkeyrefs` "pull" referenced content into the current topic. `@Conref` push and `@conkeyref` push insert referenced content into a specific location in a target topic.

For example, let's say that a topic named `Untitled1.dita` contained the following section.

```
<section>
  <title>Conref push example</title>
  <p id="p_first-para">The badge should be inserted above this paragraph. </
p>
  <p> . . . </p>
</section>
```

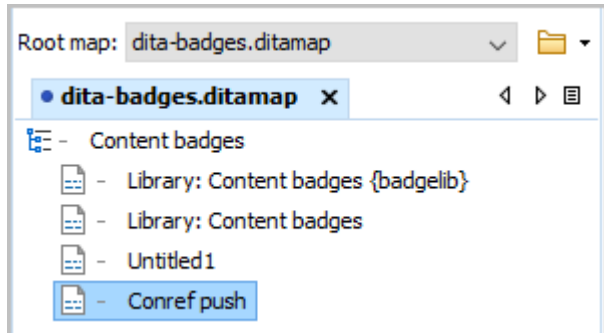
In its current state, this markup generates the following output.

### Conref push example

The badge should be inserted above this paragraph.

...

To "push" a section-level badge into a target topic named `Untitled1.dita`, you need to create a separate topic that defines what you want to push into the target topic and where it should be inserted. Let's call that topic `conrefpush.dita`.



Set the `@processing-role` attribute of `conrefpush.dita` to "resource-only" in the map so the conref push topic does not appear alongside other content topics.

All attributes:	Attribute	Value
	<b>processing-role</b>	resource-only

Setting this attribute makes `conrefpush.dita` available to the DITA map and to the DITA processor, but not visible to the customer.

To "push" a badge into `Untitled1.dita` before the section paragraph with `@id="first-para"`, insert the following markup into `conrefpush.dita`.

```
<p conaction="pushbefore">
  <image href="images/badge_cloud-compute_small_yes.png">
    <alt>This topic applies exclusively to Cloud Compute.</alt>
  </image>
  <ph> This topic applies exclusively to Cloud Compute.</ph>
</p>
<p conaction="mark" conref="Untitled1.dita#untitled/p_first-para" />
```

Before build time, you see no change in target topic content. At build time, the paragraph with `conaction="pushbefore"` instructs the DITA processor to insert the badge defined in this paragraph *before* a target paragraph element. The second paragraph specifies that target element (insertion point) as `<p id="p_first-para">`. The badge should be inserted above this paragraph.`</p>`. The processed output now includes a badge that was never inserted directly into the topic.

### Conref push example



This topic applies exclusively to Cloud Compute.

The badge should be inserted above this paragraph.

...

This would be the location where you would insert a section-level badge manually. If another team references `Untitled1.dita` without referencing `conrefpush.dita`, the badge never appears in their generated output.

As with many of these gee-whiz features in DITA, you need to weigh the benefits of implementing conref push against the complexity that it introduces. Some technical solutions create more governance problems than they are worth.

## Conclusion

---

No one said that solving complex content architecture issues in DITA would be easy. If your organization *does* need to design and implement content badging, hopefully some of the advice and best practices in this whitepaper will prove useful.

OASIS DITA Adoption Technical Committee