# What I Learned from the DITA Listening Sessions (Part 1)

By Keith Schengili-Roberts

My work for IXIASOFT includes being a company liaison for OASIS, the open standards body that produces the DITA specification. One of the groups I actively work with there is the DITA Adoption Committee [https://www.oasis-open.org/committees/tc\_home.php?wg\_abbrev=dita-adoption], chaired by JoAnn Hackos with Stan Doherty as one of its secretaries. Its goals are to help inform the DITA community on best practices and to provide general information on how to use DITA by publishing numerous white papers on various subjects [https://www.oasis-open.org/committees/tc\_home.php?wg\_abbrev=dita-adoption#expository] concerning DITA. Shortly after the official launch of the DITA 1.3 specification (which comes from the separate DITA Technical Committee [https://www.oasis-open.org/committees/tc\_home.php?wg\_abbrev=dita]), Stan Doherty asked whether this might be a good time to gauge the pulse of the DITA community as a whole before forging ahead in earnest with future DITA 2.0 development. Out of this came the idea for the “DITA Listening Sessions” where the intention was to gather DITA practitioners together and discover their attitudes about the current state of DITA. Stanley is a resident of Boston, and since he was the driving force behind the project, all of the sessions he hosted were in this area. IXIASOFT has a vested interest in understanding the concerns and issues facing DITA users in order to get a better sense as to where future development efforts should go, and so I was able to attend these sessions and aide Stanley as an OASIS DITA representative.

The three Boston-area sessions included one held in Cambridge, the second in Littleton, and the third in Marlborough. Just over 40 people came to all of these sessions, representing companies like VCE, Simplivity, Akamai, Fidelity Investments, IBM, Oracle and more. The attendees came with varying levels of experience, ranging from those who came to find out more about DITA, to those with a couple of year’s worth of experience, to those who had very mature processes and a deep knowledge of DITA. In addition to Stanley and myself, other members of the OASIS DITA committees were able to attend remotely, including Bob Thomas [http://tagsmiths.com/], Don Day [http://learningbywrote.com/], Joe Storbeck [https://www.linkedin.com/pulse/getting-know-janas-dita-consultation-development-leader-matthew-cassi] and JoAnn Hackos [http://comtech-serv.com/index.php?main\_page=index&cPath=31\_135]. There were a number of set questions for the people attending, asking what has worked and not worked well, what version of DITA they are using, and how the OASIS DITA committees can help. But after that the sessions were open to let people express their opinions and experiences with DITA.

What follows are some key points that I came away with from the sessions, along with some recommendations for the OASIS committees and DITA users.

## Madcap Flare is a Common Stepping Stone to DITA

As someone steeped in the venerable FrameMaker through much of my career in technical writing prior to DITA, I was genuinely surprised to discover how many technical writing teams are using Madcap Flare. There were certainly a few attendees who were using FrameMaker, but for those who were either contemplating the shift to DITA or had recently done so, former Flare users outnumbered those who had used FrameMaker.

Madcap Flare has the ability to create modular, topic-like content. It also enables reuse through a mechanism called “snippets”, allowing for the block-level and inline reuse of content. Both of these features are very similar to the reuse mechanisms available in DITA, which makes Flare a good stepping-stone for those thinking of moving to DITA.

Why were people thinking of moving from Flare to DITA? Apparently the program has its limits, as it is not a Component Content Management System and reportedly cannot easily handle large amounts of content. According to another attendee it does not always play well with content versioning software like Subversion, with another attendee expressing frustration over its conditional content mechanism, wishing she had the more straightforward include/exclude flexibility that DITA filtering could accomplish.

Those who were thinking of migrating content over to DITA from Flare wish that they had a guide clearly outlining the processes necessary for moving from Madcap Flare to DITA. **Recommendation:** create a white paper outlining the key differences and similarities of Madcap Flare and DITA, the advantages and how best to migrate content over to DITA.

## Confusion Between the Relationship Between DITA and the DITA-OT

The majority of the people who attended the listening sessions were using the specification along with the DITA Open Toolkit (DITA-OT). Several people expressed the desire for more capabilities from the DITA-OT, the most common requests being for more robust HTML5 output and for an easier way to craft the XSL that formats the content. Another repeated request was for more information on the inner workings of the DITA-OT, in order to make its operation easier to troubleshoot.

After several people expressed their opinions about the DITA-OT and what they wanted to see in future versions, it became clear that there was confusion about the relationship between the DITA standard from OASIS and the DITA-OT open source development project. The DITA-OT has become the *de facto* design implementation of output for the DITA standard, and for many people the two naturally go hand-in-hand. I pointed out that neither the OASIS DITA Adoption Committer nor the OASIS DITA Technical Committee controls the work done by the DITA-OT open source project. This distinction was far from clear to many of those attending the sessions, even those who have been using DITA and the DITA-OT for years.

**Recommendation:** create a white paper outlining the distinction between the two OASIS organizations that create and work on the DITA standard and what their relationship is to DITA-OT open source project. I am happy to report that this is already being worked on; there’s now an unofficial blog post [http://metadita.org/toolkit/ditaversusditaot.html] from DITA-OT developer Robert Anderson on the subject, and the DITA Technical Committee is also working on an official document on the same issue.

For those who are looking influence the direction of DITA-OT development, the best solution is to directly participate in the development process [http://www.dita-ot.org/participate], comment on any documentation issues you discover [https://github.com/dita-ot/docs/issues], or look at the list of known issues [https://github.com/dita-ot/dita-ot/issues] to see if your pet peeve is already being worked on. Anything you can do to help the small but feisty development team will improve the DITA-OT for the whole of the DITA community.

## The Transition from conrefs to keys is a Real Problem for Mature DITA Users

A fervent wish expressed by several people coming from technical documentation groups that have been using DITA for years was the desire to move to a DITA 1.2 key-based way of sharing content from their existing conref-based processes. They are sold on the idea of keys in their various forms (keys, keydefs, and conkeyrefs from DITA 1.2) as this referencing mechanism offers a more straightforward way of managing reusable content for things like product names, warehousing images and icons (i.e. “resource keys”), commonly-referenced website xrefs, topics (i.e. “navigation keys”) and more. Due to the way keys are created and managed prevents the potential “spaghetti conref” issue that can arise if conrefed content is itself conrefed. (If you are looking for a demonstration of the versatility of DITA keys, download the excellent example Thunderbird documentation set available from GitHub [https://github.com/gnostyx/dita-demo-content-collection]). Keys also offer an opportunity to simplify conditionalized content.

So what’s stopping these mature DITA implementations from moving forward? Based on the responses from the attendees, there are three main reasons for this:

* It took a long time for some DITA software tools to implement keys
* The amount of *legacy* *DITA* content based on conref reuse methods
* Relative lack of information as to the process utility of keys.

Despite the fact that keys were introduced over five years ago with the launch of the DITA 1.2 specification, there was a significant lag in the adoption of keys by some tool vendors. Depending on the CCMS or the editor being used, the implementation of keys has been late in coming. Some technical writing departments have been unable to use key-based reuse mechanisms until fairly recently. Key-based reuse is still out of reach for those technical writing teams that are not using the latest, updated tools. **Recommendation to tool buyers:** Since there are significant extensions to key-related mechanisms in DITA 1.3 (like keyscopes) new entrants to DITA should look into the history of key implementation and support within the tools they are looking to buy, and consider the development roadmap a tool vendor has for implementing DITA 1.3 functionality.

For those who have been using DITA for any length of time prior to the release of DITA 1.2, much of their content reuse below the topic level has been accomplished by using conrefs. It has become common practice at many firms to use this content with conditions in order to produce related but different document deliverables. Keys can also be used to drastically reduce the amount of conditional markup in terms of selection attributes within in publications. An example of when keys could replace conditions is when there are two or more adjacent elements with conditions that are mutually exclusive. This means that exactly one of the adjacent elements will be included while the others will be excluded. In those cases, you can have a single key that resolves differently for each scenario. A combination of keys and conkeyrefs can turn a topic into a “template” that uses keys to generate product specific content that varies only slightly from one product to the next. As complex as conrefs can get, when used with conditional markup it can makes things significantly more complex. DITA 1.2 keys mechanisms offer a way out of this quagmire of complexity. But the larger the amount of legacy conrefed content, the bigger the problem: I found it telling that one of the people who wanted to use conkeyrefs in their current implementation but couldn’t, due to legacy DITA content issues, was someone who worked at IBM. **Recommendation:** Create a white paper that describes how to architect a transition from conref-based content reuse over to keys, describing the likely pitfalls and how to avoid them.

While it was clear to some people at the sessions as to the utility of keys over conrefs, that knowledge is not pervasive. I’ve outlined some of the advantages here, but this knowledge has come from personal experience—there’s next to no information out there that specifically addresses the advantages of a key-based approach to content reuse. Several people at the listening sessions remarked that they are interested in using keys, but did not have enough information on what they could do and why they were an improvement. The focus of early articles on the subject of keys was on the mechanics of using them: there was little information as to the *benefits* of keys from a process standpoint. **Recommendation:** create a white paper outlining the process advantages of keys over conrefs rather than just the mechanics; people can find out the “how” easily enough, but some have problems understanding the “why”.

***In the next post:*** *the desire to build community, the concern that DITA is not meeting current needs for some, and the level of interest in Lightweight DITA.*

*Thanks also goes out to Bob Thomas [http://tagsmiths.com/] for his review and suggestions relating to this piece*