## DITA Strengths Analysis

**[01]** The DITA standard is based on research done on structured authoring, minimalism, content reuse, and other principles of sound information development.

**[SD: I typically attribute the architectural success of DITA to the convergence of three influences:  
1. XML/SGML best practices  
2. Minimalism-IA/best practices  
3. Object-oriented programming / best practices**

**- See attached Venn diagram]**

**[02]** DITA has a strong foundation in the theory and practice of Minimalism in training and technical information. Groundbreaking research on minimalism has been done by John M. Carroll, originally at IBM, and currently distinguished professional of Information Sciences and Technology at Penn State University. DITA continues to profit from the research of Hans van der Meij of the University of Twente in the Netherlands. The DITA Troubleshooting topic design was strongly influenced by Dr. van der Meij’s research.

**[03]** The DITA information types (Task, Concept, Reference, Troubleshooting, and Glossary) provide structure that helps to ensure consistency across content deliverables. Such consistency helps to promote quality information for consumers. **[SD: Add a nod to Information Mapping?]**

**[04]** DITA supports a topic-based approach to content, as well as a book-based approach. DITA topics can be arranged in**to** book form with structures such as parts, chapters, frontmatter, and appendices and published in PDF form. DITA topics can also be published as HTML or presented in a topic-oriented user interface such as Help topics or through dynamic publishing systems.

**[05]** DITA, based on XML, separates form**atting** from content. Topics are developed with XML markup. Format is added through processing using CSS, XSLFO, and XSLT standards. By removing formatting from the source topics, authors are freed from worrying about the appearance of the final documents, which is standardized, and can concentrate on developing high-quality content. **The impact on productivity cannot be underestimated. If a team of ten writers and editors were paid $110K FTE annually and they were required (conservatively) to fuss with formatting in their source files for 33% of their time, the lost productivity for that team over five years would be $1,815,000. Eliminating the need for fussing with page masters, layers of stylesheets, and output-specific styles improves the total cost of ownership of your publishing pipeline directly**.

**[06]** DITA experts have designed a DITA maturity model to assist new adopters in developing a roadmap for their DITA implementation. The maturity model suggests that an organization adopting DITA can start with a simple model and progress by adding capabilities as they gain experience.

**[07]** Content that is written in XML is less expensive to translate than content developed in proprietary authoring systems. XML content, separating format from content, allows translators to work only with the words. The localization services providers have no need to extract the text from a proprietary format, translate it, and then reformat it. The services providers do not need expertise on a large number of proprietary authoring systems, helping them reduce costs. Without the high cost of formatting, organizations that author in XML DITA significantly reduce production costs.

**[08]** DITA supports single sourcing of content. **Any c**ontent **that is** developed and approved once **can be** available for use in multiple outputs and formats without need to alter the **underlying** source **topics**. Source content can be structured to contain variable content that is then made to respond to different requirements in outputs, again without changing the source. Variables in DITA are implemented through the use of content reference, key references, and attributes.

**[09]** DITA has been adopted by a wide range of organizations with very different subject areas, including computer software, computer hardware, heavy equipment, medical devices, pharmaceuticals, telecommunications, aerospace, banking, insurance, human resources, marketing, and more. DITA is also used in many parts of an organization, including technical communication, product development, marketing, human resources, education and training, service and support, regulatory affairs, and more. //please add to this quick list//

**[10]** A wide variety of resources are available for learning DITA, including books, articles, webinars, workshops, and conferences.

**[11]** DITA is supported by an increasing number of open-source and proprietary tools, including tools that support authoring, content management, publishing, and translation. The variety of options enables DITA adopters to avoid being locked in to one product or to a proprietary solution. A strong consulting community provides education and assistance in implementing DITA successfully.

**[12]** An **increasing**  number of  **DITA-compliant authoring tools are designed to** support both **entry-level and expert content developers**. These tools, in addition to DITA constraints, specializations, templates, and quality-assurance capabilities, ensure that DITA can be successfully introduced to authors who are reluctant to learn XML markup. In many cases, authors are unaware that they are produced valid DITA content.

Add: Significant body of research from MIT, IBM et. al. over many years

resulted in DITA Bob Doyle’s history of DITA in in dita.xml.org Don Day, Dave Schell article on history of DITA. <http://dita.xml.org/book/history-of-dita#comment-3981>

**[13]** For architects, authors, and integrators, DITA offers a world of options. Organizations adopting DITA have considerable flexibility in selecting tools that are available at every level of investment. From open-source processing environments to inexpensive editors to mid-range Component Content Management Systems (CCMS) to high-end enterprise content management solutions, you determine what mix of DITA-compliant tools would best support your business needs – today and tomorrow. You can deliver “best of breed” solutions for your organization and for your customers without getting locked into vendor-specific or platform-specific solutions. [Reworded from Don Day’s feedback]

**[14]** As DITA becomes the “standard content development standard” for many high-tech segments, the population of writers, architects, and tool smiths trained in DITA has grown enormously. Recruiting from this pool of DITA-savvy candidates saves your organization training costs, ramp-up time, and long-term inconsistencies in content architecture. [Reworded from Don Day’s feedback]

**[15]** The collective community knowledge about the DITA standard and how organizations can deploy it successfully is significant. The numerous user group sites, blogs, mailing lists, LinkedIn groups, and open-source homepages dedicated to DITA worldwide are not hidden behind logins or customer support licenses – they are available to everyone, monitored informally by DITA experts, and open to new contributors with great ideas. Organizations are more readily able to innovate with DITA because so many of the technical issues that they might encounter along the way have been recognized, researched, and documented. [Reworded from Don Day’s feedback]

**[16]** DITA was designed to be flexible in its XML grammars (markup rules) and extensible in the way you can customize them. With DITA 1.3, you can choose to implement your authoring shells or customizations using one of three open-standards for XML grammars: [RelaxNG](http://relaxng.org/) (<http://relaxng.org/>), Document Type Definitions (<http://www.w3.org/TR/REC-xml/>), or XML Schema (<http://www.w3.org/standards/xml/schema>). They each have their relative strengths and weaknesses – you choose. In terms of customizing DITA maps, DITA topics types, elements, attributes, or metadata, you can choose to develop custom objects by extending their structure from other existing DITA objects (structural specialization), by disabling select components from an existing object (constraint specialization), or by developing entire new libraries of semantic elements from scratch (domain vocabularies). The incremental investment in developing and maintaining these customizations is quite modest because any new object that you customize (subclass) from an existing DITA object derives (inherits) the majority of its logic from its parent object. As the complexity associated with customizing DITA objects and the DITA environment in general diminishes, the role of creating new objects and vocabularies shifts from the OASIS DITA Technical Committee to the companies implementing DITA. By design, the DITA 1.3 standard is a foundation and starting point for your journey, not a shrink-wrapped set of targeted designs and templates.

**[17]** From the point of view of tools development and integration, DITA implementations have adopted a plug-in architecture that works with multiple frameworks. To add a domain vocabulary of elements relevant to your particular industry, you never need to touch the default RNGs or DTDs that ship with OASIS DITA. You simply point your customized objects at the default OASIS DITA objects and specify the variations that you want in your customized objects. When you wrapper those customizations as a plug-in, you can easily deploy, update, or deprecate them across departments, divisions, or subsidiaries within your company. When a new release of DITA or of a DITA-compliant framework is released. your plug-ins should work with little or no modification. The combination of object inheritance and a plug-in architecture makes the DITA tool environment both flexible and cost efficient.

**[18]** If writers and their organizations can share and reuse content, the effects on quality, consistency, and localization costs are significant. Many DITA implementations recoup their conversion and retooling costs within 12-18 months through localization savings (translating fewer words). To ratchet up the amount of DITA content that can be reused, you have many options. First, you can target the *granularity* of the information that you plan to reuse – maps, topics, sections, common blocks, images, and phrases. Second, you can target the *directionality* of reusing content – referencing content into your current object (conref “pull”) or referencing content into some other object (conref “push”). Third, you can program *conditionality* into your environment, i.e. using DITA filtering rules to include content in one pipeline but exclude it from others. Over time, DITA objects begin to contain more content that gets “built” for any one deliverable. Lastly, you have fine control over *context*, i.e. how these reusable chunks are referenced (indirect keys, or direct URIs), encapsulated (hard linking, no linking, computed linking), or scoped (keys, filtering).

**[19]** Semantic markup tends to be one of the least-appreciated strengths in DITA. In authoring environments oriented toward the presentation of content, writers tag their content with elements such as <p> (paragraph), <li> (list item>, or <b> boldfaced. Looking at the tags, it is impossible to determine what type of content should actually be captured by those tags. In contrast, DITA makes available a baseline set of tags that describe the meaning (semantics) of the content captured by those tags. To capture a sequence of GUI menu selections, for example, you would enter <menucascade><uicontrol>File</uicontrol><uicontrol>Export</uicontrol><uicontrol>to PDF</uicontrol></menucascade>. To capture a formal step in a procedure, I would enter <step><cmd>Select the node in your Federation.</cmd><stepresult><p>Network Manager displays the following settings page.</p><image href=”screen1.png”/></stepresult>. In the DITA 1.3 troubleshooting topic type, I would describe a possible solution to a problem with the following markup:

<troubleSolution>  
 <cause>  
 <title>Corrupt routing table</title>  
 <p>Your control server has a corrupt routing table.</p>  
 </cause>  
 <remedy>  
 <responsibleParty>IT</responsibleParty>  
 <steps-unordered>  
 <step>  
 <cmd>Enter this command to regenerate the table.</cmd>

<info><codeblock>regen-routing-table</codeblock></info>  
 </step>  
 </steps-unordered>  
 </remedy>  
</troubleSolution>

The element names and (sometimes) the sequence in which you enter them encourage writers to focus on the function of the content within each element and across the larger context of the topic. As the result of semantic markup, organizations tend to get more focused content at the element/section level and more consistent content across topics.

**[19]** The "A for Architecture" in DITA means that it is not specific to any one industry or type of information. A basic DITA topic can, in fact, be nested and used to create journal- or even book-length discourse with only the expected markup for such use. In short, DITA is a framework that supports adaptation to many kinds of structured, semantically-identified content, both document- and data-oriented. [Don Day]