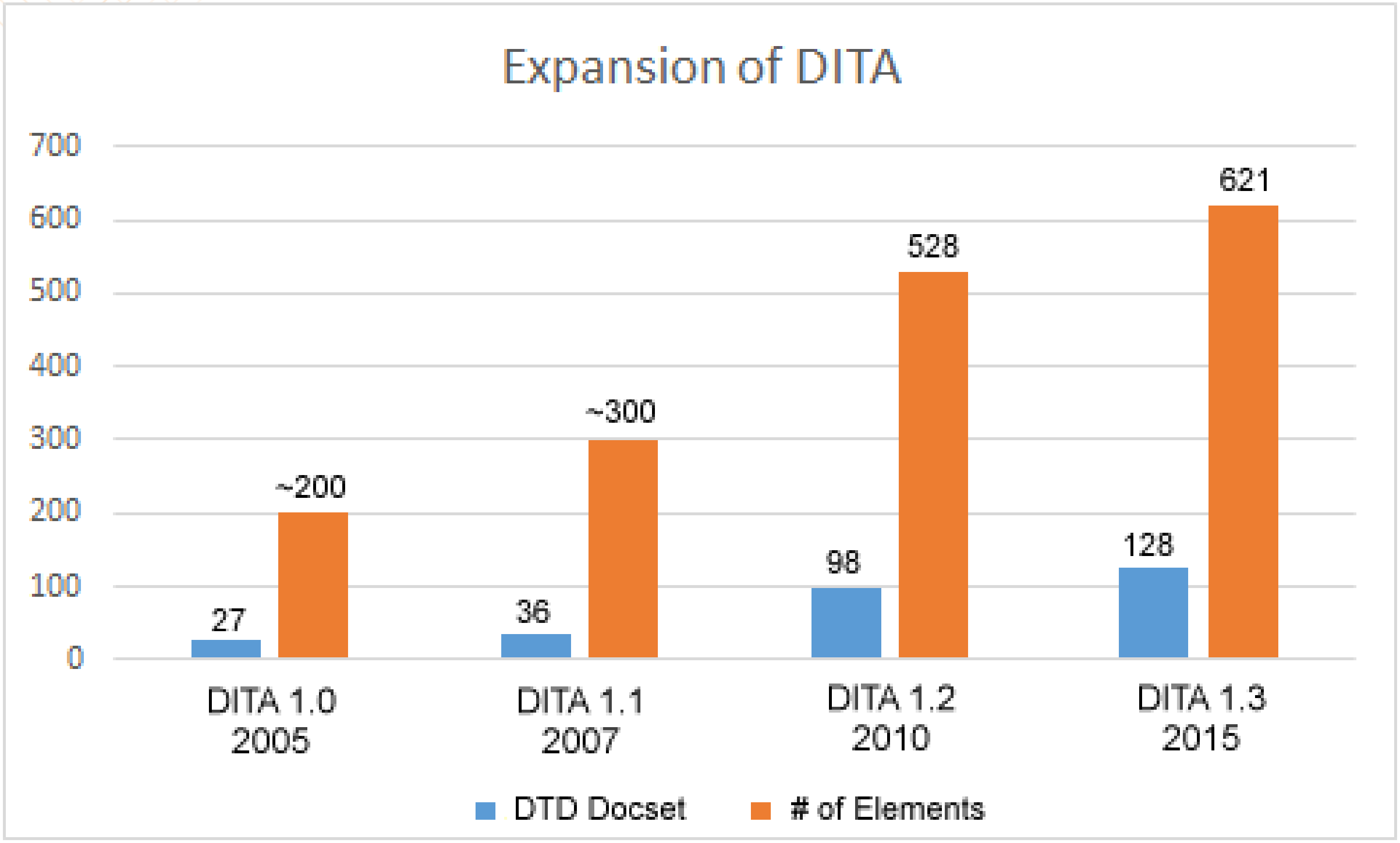
The DITA Iceberg

Leigh White, DITA Specialist

DITA Europe, November 2016

IXIASOFT

Forget big data…we have big DITA!



So where has that brought us?

IXIASOFT

Added in DITA 1.1

- bookmap
- glossentry
- abstract
- foreign, unknown
- data
- indexing improvements: see, see-also, page ranges, and sort order
- Specialization support for new global attributes
- Conditional processing profiles

Added in DITA 1.2

- keys
- constraint modules (more on this later)
- new glossary elements
- conref push/range
- general task vs. strict task
- miscellaneous new elements
  - text, mapref, sectiondiv, etc.
- Learning and Training specialization
- subjectScheme
- machineryTask topic

Added in DITA 1.3

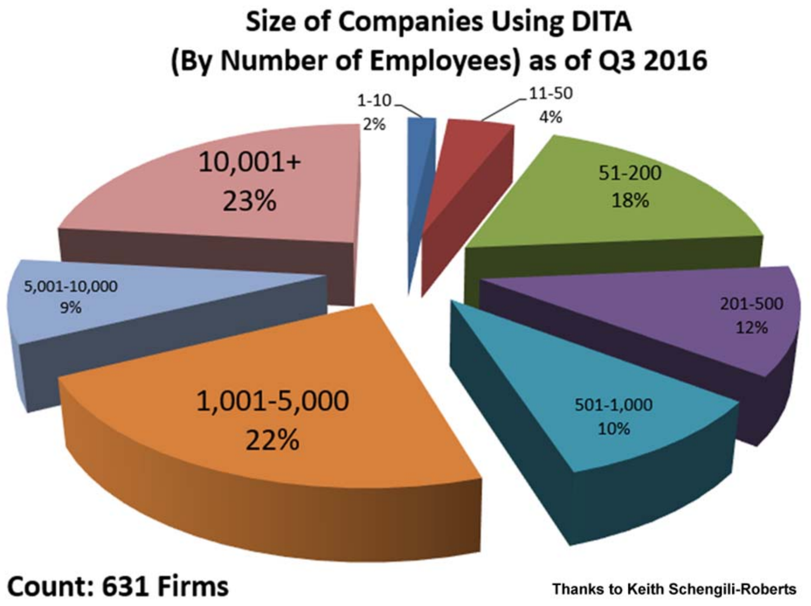
- scoped keys
- branch filtering
- troubleshooting
- XML Mention domain
- context-sensitive help
- release management
- Relax NG
- SVG integration
- MathML integration

What follows is my perception, not **entirely** without a factual and experiential basis…

IXIASOFT

90/10?

- Are we doing 90% of development work for 10% of user base?
  - Definitely not that much, but…
  - DITA usage almost equally split between companies <1000 and >1000 employees (based on survey of 631 companies).



IXIASOFT

Company size → IA/toolsmith availability?

- >55% of doc team members in companies >1000 employees do not have the traditional "tech writer" roles (i.e. might be IA's or toolsmiths instead)*
- ~30% of doc team members in companies <1000 employees do not have the traditional "tech writer" roles*
- Can we assume that
  - Large teams more likely to have an IA/toolsmith
  - Small teams more likely not to have an IA/toolsmith
  - This is not news!

* Again, thanks to Keith Schengili-Roberts for the number-crunching

DITA 1.3 feature implementation

- That leaves roughly half of DITA teams likely not to have a dedicated IA/toolsmith
- What DITA 1.3 features are these teams likely to implement?
  - Not likely:
    - Release Management (requires plugin dev resources & budget)
    - Context-sensitive help (ditto)
    - XML Mention domain (unless their product is XML-based)
  - Maybe:
    - Scoped keys (if they have a resource who can manage it)
    - Branch filtering (ditto)
  - More likely:
    - Troubleshooting

DITA 1.3 feature implementation [2]

- So a lot of proposal evaluation, approval, spec development, OT development and DITA documentation was done for features below the waterline
- And this complexity is present for everyone, not just the power users
  - If you don’t want it, you can’t “hide” it easily

Acknowledgement of complexity

- Specification available in three editions:
  - Base
  - technicalContent
  - All-inclusive
  - (But this implies ability to **easily** use just Base elements, which is not really the case)
- Series of OASIS Adoption Committee articles to explain features
- Lightweight DITA (LwDITA) (more in a minute)
- Tools to simplify the authoring experience

LwDITA

- Not necessarily meant as a simplified authoring environment
- Designed to be “entry point” (or maybe pivot point) for HTML5, Markdown
- Adequate for content creation otherwise?
  - For beginning DITA authors
  - For casual contributors
  - For groups with basic structured content needs
  - What if you need more than LwDITA but less than the full tagset?

Constraints

- Are not the answer!
- Introduced in DITA 1.2
- Acknowledgement that, “Hey, we have a $%#&-ton of elements here and many (most?) people aren’t going to need them all.”
- In real life, how often do you take the same approach?

IXIASOFT

Constraints are not the answer

- I'll buy a pickup truck:

- But I don't need a truck bed, so I'll cut that off:

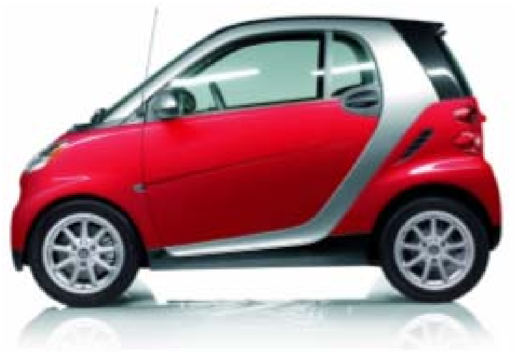
- And I don't need a high profile, so I'll lower it:

Constraints are not the answer

- I don't need that big V8 engine either, so let me swap it out for a V4:

- And, now at last, I have the perfect car for my city driving and parking!

- Why not just buy



to begin with?

A fork in the road?

- Standard DITA
  - More robust than LwDITA but still pre-constrained list of most commonly used & accessible elements
  - No “special interest” elements
- Advanced DITA
  - The whole ball o’ wax
- **Not** achieved via downsizing using constraints!
  - Au contraire, start with Standard and upsize to Advanced
    - With easier mechanisms than are currently available…plugins?
  - Interchange? Still doable!
    - Authoring environment: Standard
    - Production/validation environment: Advanced

The good news

- DITA 2.0 is moving in this direction!
  - Freed from requirements of backwards-compatibility
  - Elimination of redundant elements
  - Elimination of some “special interest” domains
- What else can/should we do?
  - Have I misspoken or misrepresented?
  - What else is in the works for 2.0?
  - How can we make DITA accessible out of the box to even the smallest, non-technical doc teams?

IXIASOFT

Questions/Comments?