

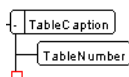
Automating conversion to structured FrameMaker

Our client creates and maintains documents that must follow United States military specifications (“mil-spec”). To standardize the authoring process, we developed a structured authoring system that also produces XML for content exchange with other agencies.

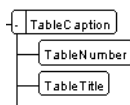
The client had thousands of pages of documentation and limited resources to devote to conversion. In addition, access to certain documents was highly restricted. We created a push-button solution based on FrameScript scripts that *reduces the conversion time by 70%*. The author applies structure to FrameMaker documents in minutes, not hours.

Formatting in the legacy documents presented several challenges, including the following:

- ❖ Some irregular formatting needed correction. For example, the table number and title were both formatted with the table number paragraph tag. After conversion, the number and title were wrapped in the TableNumber element (shown below on the left). Our script formatted the table title correctly, resulting in a valid structure (shown below on the right):



Before



After

- ❖ Some formatting required manual cleanup in the structured documents. Our script updated most cross-references; where author intervention was required, the obsolete cross-reference was replaced with an Error indicator and a console message:

See **ERROR** for more information.

```
Addendum # cross-reference format is not in the find or change list.
C:\originals\test.fm
```

Push-button conversion

To clean up and structure the legacy documents, the author runs one script that performs the following tasks:

1. Imports formats from the FrameMaker template, removes overrides, and applies the correct template formats in their place.
2. Imports the element definitions and applies the conversion rules to create a structured document.
3. Corrects common invalid elements (for example, table elements that couldn't be mapped to unique paragraph tags in the conversion rules).

Typically, the author would spend hours manually performing these tasks. Using scripts, the author can structure a book in minutes, and the scripts do a better job finding and repairing inconsistencies.

Scripts are easy to customize. Authors might discover odd formatting in a manual during conversion. To fix the problem, they can create or modify a script in a text editor.





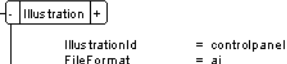
The client purchased a copy of FrameScript for each author so that authors can convert their own documents. This increases the conversion rate and illustrates the importance of template compliance. If the author has applied template formats consistently, the structured documents have fewer invalid structures than files with customized formatting.



scriptorium.com

Copyright © 2005 Scriptorium Publishing Services, Inc.

The following table describes some of the scripts:

Problem	Solution
Scripts for unstructured files	
Untagged formatting Untagged character formatting isn't wrapped in the correct elements after documents are structured.	Script searches for untagged character formatting in unstructured files and applies the correct character tags.
Hazard symbol variables To create a hazard symbol, such as  , authors often type in a letter and apply a custom font instead of inserting a user variable. In the structured files, the letter converts to plain text: <div style="margin-left: 40px;">  </div>	Script searches for the Hazard font applied to text and inserts a user variable based on the typed letter. The text is then wrapped in the Hazard element with the corresponding Symbol attribute. <div style="margin-left: 40px;">  </div>
Capitalization Headings are capitalized inconsistently—mixed case or lowercase—which results in mixed case headings in the table of contents.	Script updates capitalization in heading paragraphs. Exception words (articles or acronyms) aren't capitalized.
Scripts for structured files	
Formatting tables Some table elements don't structure correctly because the elements can't be mapped to unique paragraph tags in the conversion rules.	Script searches for specific table elements and applies the corresponding elements.
Illustration attributes In graphic elements, the referenced file name and extension are stored in attributes. This information cannot be extracted during the conversion process, so the attributes are invalid in the structured files. <div style="margin-left: 40px;">  </div>	Script finds each graphic element and populates the attributes automatically. <div style="margin-left: 40px;">  </div>
External cross-references When documents are structured, the files are saved in a new folder. The external cross-references still point to the original folder.	Script asks the author to type in the original and new folder names and then searches for and replaces the folder names.

Conclusion

A script-based system has several advantages:

- ❖ The client can convert confidential documents without extensive training on adding structure.
- ❖ They can modify and create new FrameScript scripts when necessary.
- ❖ Ultimately, the client can apply structure to FrameMaker documentation 70% more quickly in our FrameScript-based conversion system than without the scripts.

The expense of legacy document conversion can be a significant roadblock for companies that need to implement structured authoring. Scripting requires analysis and development time, but those costs pale compared to the cost of manual conversion. If you have more than 500 pages of content to convert, we recommend scripting and other automation.

