An effective and automated publishing process to improve user interface style guides

Martin Lugmayr; Johann Schrammel; Cornelia Gerdenitsch; Manfred Tscheligi

CURE – Center for Usability Research and Engineering
Modecenterstrasse 17/Objekt 2, Vienna, Austria
{lugmayr, schrammel, gerdenitsch, tscheligi}@cure.at

Abstract

Style guides have become an important and common way to improve and standardise development of user interfaces. However, there are several well-known problems on using style guides. Having these problems in mind we present an effective and automated publishing process. First we will introduce a role-based approach to model style guides. After that we will focus on the steps of the publishing process and describe them in detail with their outputs. By that we want to focus on the practical and theoretical advantages of our methodology and their limitations. In summary, this paper will describe in detail how mentioned techniques and components work together and how we build up a useful publishing process for adaptable and usable style guides.

Keywords: user interface style guide; DITA maps; user centred design; usability

1. Introduction

User interface style guides are a central and important element for developing graphical user interfaces (GUIs). With their aid it is possible to guarantee consistency (e.g. menu guidance, Look and Feel) between different applications [2], to provide a high quality human-computer interaction [5] and to simplify interdisciplinary and multinational collaboration in developing GUIs [4].

There are several typical problems that occur in developing, implementing and using style guides that are related to the commonly used traditional publishing process. Scientific literature and practical experiences

illustrate that the reason for many problems is related to the preparation and representation of information in a style guide. Wilson [6] reports issues on updating, bad usability, insufficient indexes and others. Similar problems are formulated by [4] and affect for example style of preparing materials, media (paper-based vs. online) and the complexity in practice of style guides. Those authors also pointed out that people who use style guides want to get the information they look for structured and fast.

Reasons for those problems that exist in using style guides are e.g. formulated by [6]. The author lists thirteen reasons that can cause problems in using style guides. Five main reasons concerning the design and implementation of the style guide are formulated as follows:

- Extent of the style guide: Although guides should be very easy to understand, complete style guides became very big.
- Possibilities of updating: The question is how to distribute updates to the style guide. Some try to put the guide online, but in this case you still need to alert people to do changes.
- Bad usability: Users often do not understand the guides and also have to look for the information they need very long.
- Insufficient indexing: There are not enough index terms (which are additionally not sufficient) and there is also a poor use of cross-referencing.
- Too much formulated text: There is too much formulated text instead of integrating screenshots or bullet points.

Other problematic areas are the complexity of the style guide and that style guides are laborious to use [4].

In this paper we want to present a new style guide approach that gives a solution for many of those problems. We present style guides based on DITA maps. First we want to describe the role-based approach, the publishing process and the technical implementation of the guide. Afterwards we provide a demonstration of our methodology. In the discussion section we work out positive and negative aspects of our work.

2. Methodology

Taking above mentioned problems into account, the aim of our research was to establish a new method to develop, publish and maintain style guides.

To achieve our goals we use two different approaches. First we want to reduce complexity of the style guide by introducing role-based style guides.

Second, our aim was to improve and automate the update process by using DITA.

Within our publishing methodology typical problems can be avoided, and as a consequence we expect an increase in the acceptance of style guides in the practical context.

The next step is to evaluate our approach in user trials and interviews with real users of the developed style guide.

2.1. The role-based approach

The main idea of the role-based approach is to split up the information into small logical units that are maintained centrally, to determine the relevance of these information units for the different user groups and output formats and to produce tailored documents automatically. Based on the modelled scaffold and with the help of automated transformation processes, specific documents for specific users and publication formats are generated. Consequently we can make sure that different user groups get just the information they need and we can avoid providing irrelevant and unnecessary content.

Regarding the implementation of our particular style guide, the following three roles were defined to cover the different needs and requirements of the different user groups:

1. User Interface Designer
2. Developer
3. Library-Developer

Corresponding to these three roles we split up the content of the whole style guide and assigned each logical unit to at least one role. Due to the fact that we designed the roles according to an extensive analysis of internal processes and division of work of our project partner, other roles will probably be more feasible for other applications.

Using the tagged content for each role, a tailored style guide can be generated automatically. Thus, each user will get the required information for his/her role. Nevertheless, each user is given a chance to explore the whole style guide and not only the filtered parts. This will provide an overview of all topics additionally.

2.2. The publishing process

In addition to the aim of reducing complexity of the style guide, our second approach is to avoid the mentioned typical problems that occur when using a conventional production and publishing process for style guides. The technical implementation focuses especially on an efficient publishing process, which supports the user in getting the required information.

Contrary to conventional style guides which are monolithic systems, we used a modular concept. Therefore, we split the style guide and extracted hundreds of small topics from chapters, sections and subsections. Consequently, a topic is our basic information unit.

2.3. Technical Implementation

According to our idea of structured content, we required technology which provides both a reliable and automated publishing process and the ability to handle tagged content units.

DITA (Darwin Information Typing Architecture) achieves these requirements. Embedded Ant and Batch scripts for automation as well as a topic-based structure provide the required functionality. Thus, we designed the publishing process of our style guide based on DITA specifications.

The mentioned topics are stored in XML-files and build the basis for further data processing in DITA. DITA Open Toolkit[1] is an open source implementation of DITA and thus the main component of our publishing process. DITA Open Toolkit – an XML based framework – is used for generating, distributing and reusing technical information and we will show how the capabilities of DITA Open Toolkit fit for our aims of structure, changeability and automated publishing process.

[1] http://dita-ot.sourceforge.net/

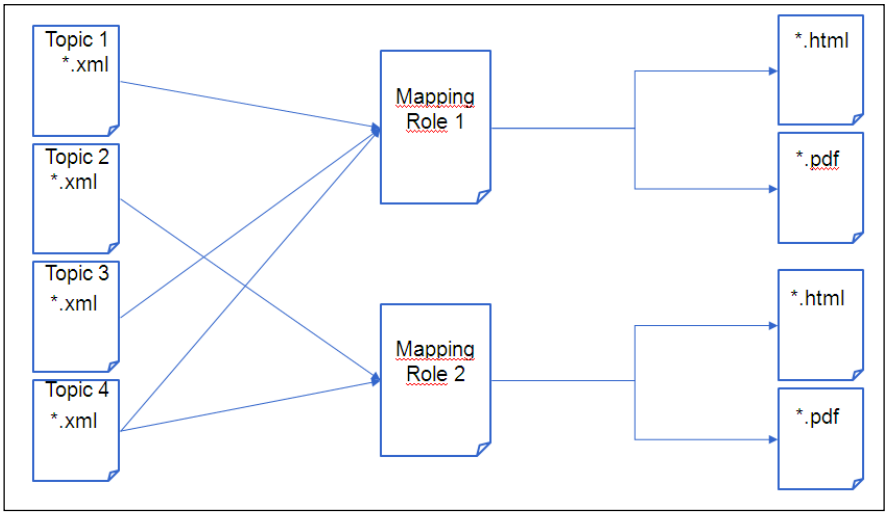

Figure 1: Schematic representation of the automated publishing process

DITA maps are a main concept in DITA and are mainly used to define the structure of a style guide document. In DITA maps all necessary topics are listed hierarchically by using a reference to the corresponding XML files, as shown in Figure 1. According to the structure in the DITA map, DITA Open Toolkit creates table of contents and structured documents automatically.

As DITA Open Toolkit implements the DITA specifications, we adopt all the useful characteristics of DITA like various output formats (HTML, PDF, MS HTML Help) from a single XML repository.

Due to the fact that DITA Open Toolkit is based on Java, XML and XSLT, we got platform-independency on the top.

The role-based approach is managed in DITA Open Toolkit by using XML attributes. Basically, each topic is labelled by an XML attribute that defines for which role it is important. On using different values for different roles it is possible to filter out the unnecessary topics. Having integrated this functionality in our publishing process, we make sure that each user gets the information s/he needs, according to the role-based approach.

Corresponding to the aim of an automated publishing process we used Ant and Batch scripts for automation. At this point of process it is defined which output formats should be created. In the background, DITA Open Toolkit uses different XSL transformations for different output formats (e.g. HTML, PDF, MS HTML Help Files, RTF). Ant scripts enable an easy and fast publishing process, which means that it is possible to build all defined output formats for all defined roles just by a double click.

A WYSIWIG XML Editor is also an essential part of our publishing process. So, defined administrators get the possibility to apply necessary changes to the style guide. Providing such update mechanisms is very important for acceptance of style guides [6]. According to this, we also introduced a SVN (Subversion [2]) repository for interoperability and traceability of changes.

2.3.1. Steps of the publishing process

To give an overview of how the publishing process is used, we describe it in the following step by step. Central aim was also to keep the update process as simple as possible.

1) Check content out of version control (Subversion): At first it is necessary to fetch the content from a central version control repository, due to tracking and security considerations.
2) Changing content in WYSIWIG XML Editor: Defined "content administrators" can insert, update or delete content. They have to assign the topics to one or more roles. That means they have to be familiar with underlying role approach and have to decide which content relates to whom.
3) Check content in to version control: Changed topics have to be stored back to version control repository, so, more than one person can administer the content of the style guide.
4) Start batch process: Re-building of the style guide and its various files can be done regularly or – if required – by executing a batch file.
5) HTML, PDF, Help file for defined roles will be created automatically: The user can always work with latest version of the style guide.

3. Live demonstration

In the following we present some output examples and details of the style guide we developed to demonstrate the advantages of our approach of the publishing process.

[2] http://subversion.tigris.org/

As discussed before, we implemented an automated publishing process which allows content distribution to various output formats for different user groups (roles). Figure 2 shows at the left side a snippet of the section overview of the PDF style guide and at the right side the HTML equivalent. As you can see, there is no difference in content. Just the appearance differs slightly due to technology constraints.

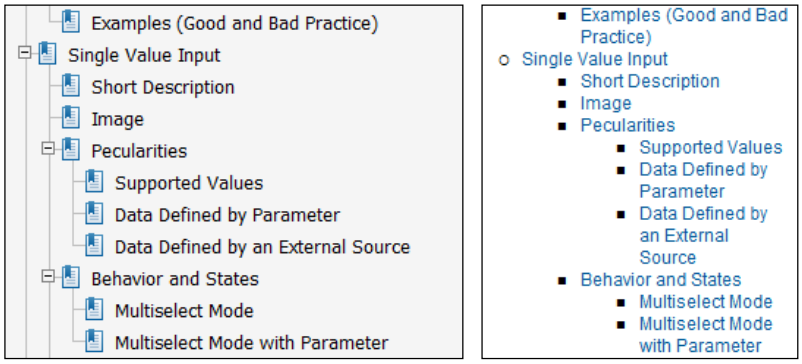

Figure 2: Section overview of content in PDF (left) and HTML (right)

Of course, not only the structure but also the content has to be the same in different output formats. Figure 3 and Figure 4 demonstrate the advantage of storing information in XML. Due to this, it is possible to generate various different output formats automatically without additional efforts. As a result, we can provide the same content for the same role in e.g. HTML, PDF, MS HTML Help. Thus, users of the style guide can switch between e.g. HTML and PDF without loss of orientation.

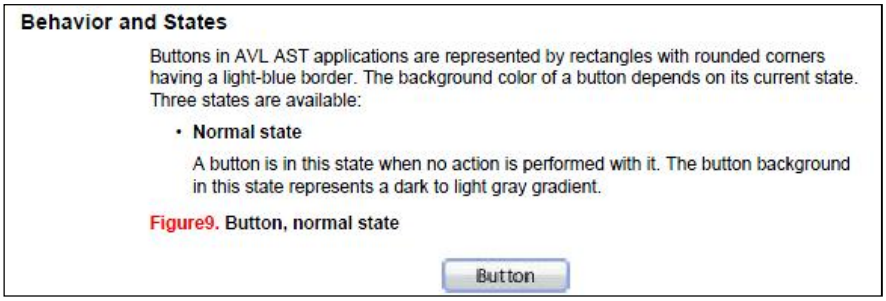

Figure 3: Content presented in PDF

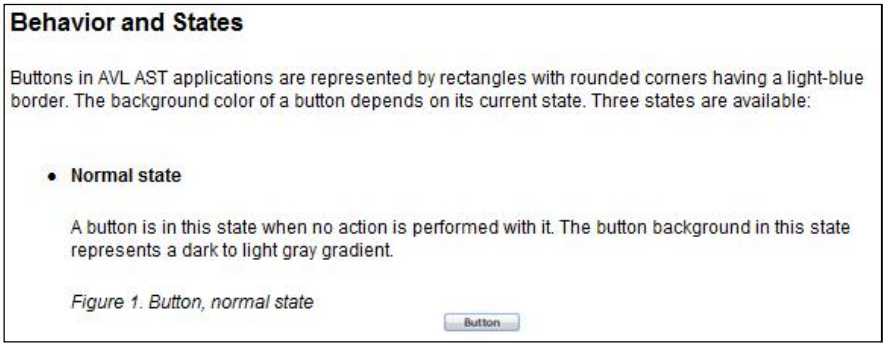

Figure 4: Content presented in HTML

According to the role-based approach we modelled the information about e.g. "Buttons" differently for the role "library developer" and "developer". As you can see in Figure 5, "library developer" will get information about the "Behavior and States", because this role is responsible to implement new buttons (or other user interface elements) in a consistent way.

Figure 5: Section “Button” for role “library developer”

As shown in Figure 6, we defined the role “developer” as not in need of the subsection “Fonts” and “Behavior and States” of buttons. The reason for this is that it’s not relevant for a developer how the button will appear, because s/he has just to use it and implement the functionality. Due to this the developer gets information about “Usage Guidelines” and “Examples”.

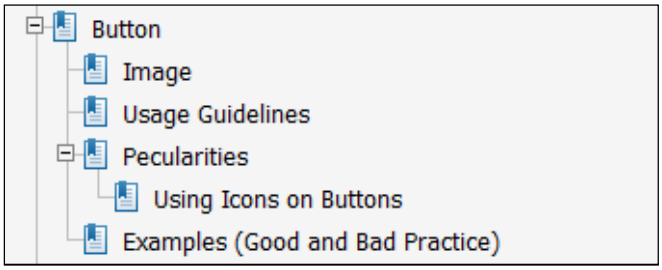

Figure 6: Section “Button” for role “developer”

4. Discussion

Developing and implementing a style guide based on DITA maps will lead to several advantages.

- Firs, it is possible to actively integrate the user in the development process of the style guide, which will consequently lead to a more flexible and easier service of the style guide.
- Second, the consistency of the style guide will be increased and for that it is much easier to compare different style guides or versions of style guides.
- Third, we want to argue that through the integration of DITA maps regular services will be done easily and fast, because it is possible to quickly react on new requirements and update styles. DITA maps allow a direct connection on online-publishing channels and for that changes came fast to the specific user and without expensive distribution costs.

Summarizing our arguments, using DITA maps can deal with traditional problems in implementing style guides.

As mentioned before, DITA maps allow integrating the user in the developing process and increasing the commitment of the end-user. For further development we suggest specifying the introduced DITA maps in a

more individual way and adapt them to specific user groups. For that we suggest formulating a scaffold, where it is possible to use it for contents of specific issues (e.g. developing an e-commerce portal). We assume to individualize the style guides as much as possible to provide the appropriate information.

We suggest integrating users in the development process as they can give valuable feedback and stimulations in developing the styles (e.g. through discussion boards). This will lead to a higher commitment on the style guide and allows adaptive improvements of the style guide within the development circle.

Further research and development should also concentrate on the output format of the DITA Open Toolkit. An opportunity in this context would be to provide code snippets as well as to formulate existing patterns and design. We want to point out that this will lead to an optimized designing process of the user interface.

5. Conclusions

Finally, development and initial setup of a style guide using both the role-based approach and the automated publishing process with DITA are more time-consuming than writing a common style guide. But in the long run there are several advantages which have to be taken into account also.

The most important advantage is the updating process of this approach, as changing content doesn't lead to extensive re-design and expensive distribution costs. Thus, the content of the style guide can be held up-to-date easily and the user gets more current information.

In addition, the role-based approach provides more relevant information to the user, because just the content that is most informative and useful for a particular role is presented. But, of course, the quality of relevance of content depends on an extensive analysis of required roles at the beginning of the style guide development process.

Both together improve style guides essentially and avoid the common problems like too large style guides and missing updating possibilities.

Notes and References

[1] Gale, S. A Collaborative Approach to Developing Style Guides. Conference proceedings on Human factors in Computing Systems, April 13 - 18, 1996, Vancouver Canada. ACM Press, pp. 362-367. 1996

[2] Henninger, S. Creating organization-specific usability guidelines. In CHI '97 Extended Abstracts on Human Factors in Computing Systems: Looking To the Future, Atlanta, Georgia, March 22 – 27. 1997

[3] Quesenbery, W. Building a better style guide. Proceedings of UPA 2001.

[4] Schemenauer, P.J., Pawlick, C. Evaluating Guidelines for Writing User Interface Text. SIGDOC'07, October 22-24. 2007

[5] Willems, P., Verlinden, J., Troost, P.J. Towards a Framework of Methods on UI Style Guides. CHI 2000, 1-6 April, p.131. 2000

[6] Wilson, C. E., STC Usability SIG Newsletter: Usability Interface, Vol 7, No. 4, April 2001 (http://www.stcsig.org/usability/newsletter/0104-style.html) (January 2010).