Classification of descriptions used in software and interaction design

Georg Strom

DIKU, University of Copenhagen, Universitetsparken 1, DK-2100 Cph. O., Denmark - georg@diku.dk

Background

A large number of different types of descriptions – or genres – are used in interaction design and software development. Each genre is suited for capturing some aspects of an interaction design or software development, so designers who choose the wrong genre may inadvertently leave out essential information or spend time capturing unnecessary information because it appears to be required in the genre they are using.

I will therefore present a classification that makes it easier to get an overview of the most common genres and the purposes for which they may be used.

Classification principles

The most important aspect of a text is what is expressed within it. Therefore I have chosen to base the classification of genres on characteristics that determine what may be expressed within each genre and the types of expression that each genre invites the user to do. In order to make a reliable classification I have also chosen to use characteristics where it is possible to determine unambiguously whether they are present in a specific text. I identified four pairs of characteristics that fulfill both criteria:

Flexible versus structured: A structured text consists of a given set of elements – often sections – that in most cases shall occur in a certain order.

Static versus progressive: A progressive text describes events that occur at different times and in general in the order in which they occur. I use the term progressive and not the term narrative – or story – because a text may be progressive without being narrative in the literary sense of the word (as described by White 1981).

Generalized versus specific: A specific text describes something that appears to be a description of an event that happens at a specific time and place, whereas a generalized text describes a concept based on a number of events with some similar aspects.

Open versus formal: In a formal genre, the wordings and grammar shall follow narrow and precise rules. I do not define a formal text as one that can be proved logically right or wrong; As Naur (1995) demonstrates it is perfectly possible to have proofs without formalization, and it may be impossible to prove a formal text right or wrong without some extraneous information about what is to be proved and how.

I found that the following widely used characteristics were unsuitable for classifying genres used in interaction design:

Fiction versus non-fiction: We are used to divide texts according to whether they describe real or imagined events. However, the same genre, for instance scenarios, may be used to describe fictitious and real events, which makes a classification into fiction and non-fiction less relevant.

Voice – the feeling a text conveys of the writer as a human being who believes what he or she writes - is often what gives the reader the strongest immediate impression (Elbow 1981). However, most technical texts have very little voice at all, and even when the voices of the texts are more varied, it is difficult to make a reliable classification based on them.

Purpose: Naur (2005) classifies texts according to whether they are intended to be descriptive or prescriptive, and we often consider the purpose of a text one of its most essential characteristics. However, if the purpose of a text is not stated explicitly, it may be difficult to determine. In some cases a text may even be regarded as prescriptive at the beginning of the development process and later be considered descriptive, when it is added to the documentation of the developed software.

Creative versus non-creative writing. According to Cheney (2001) creative writing uses dialogue, and descriptions of emotions, settings and places, whereas non-creative writing only describes the events that occur. However, HCS – Human-centered stories – is the only genre used in interaction design that may be regarded as creative, which makes a classification based on creative versus non-creative writing less relevant.

I identified fifteen different genres that are used during software development, and based my characterization of them on samples in my possession and on available literature, in particular a course book covering most of them (Strøm 2005).

		Flexible		Structured	
		Static	**Progressive**		**Static**
General	**Open**	Prototype	Scenario	*Operation:* Inst. guides User guides Use cases	*Goals:* Param. list Patterns Prototype Req. spec's Standards
	Formal			*Design:* Pseudocode	UML
Specific	**Open**	Persona	*Situations:* HCS Scenario	*Use studies:* Reports	Persona Reports

Figure 1: *Karnaugh map showing the different genres and groups*

The texts in most genres consist of a main or dominating element and an introduction and other explanatory information. I classified the fifteen different genres according to the characteristics of the main elements of them, assigned logical values to each pair of characteristics and constructed a Karnaugh map – a graphical overview often used to organize logical values – using the four pairs of characteristics.

Results of the classification

The Karnaugh map made it possible to divide all genres into five groups, where all genres in four of the five groups may share all four characteristics, and the two genres in the last group have three out of four characteristics in common. In addition, it appears that all genres in each group can be used for similar purposes - to describe one aspect of an interface or its use - during interaction design or software development. See figure 1.

Descriptions of the goal of a development process

Structured, descriptive, generalized and open:

Parameter list: Defines possible settings, input or output variables for the software to be developed.

Patterns: Describes possible specific needs and solutions that already have been designed to fulfill them.

Prototypes: Shows the interface or part of it (a prototype is not necessarily structured, it may also be flexible.)

Requirement spec's: Describes characteristics that shall be fulfilled by the developed software, but not necessarily how they shall be fulfilled.

Standards and guidelines: Are more generally defined characteristics that shall be fulfilled. They may be applicable to a range of projects or interfaces.

Patterns and prototypes require that the specific way a need is fulfilled is described, whereas requirements, standards and guidelines may describe only the need to be fulfilled.

Detailed descriptions of the operation of an interface

Structured, progressive, generalized and open:

Installation guide: Describes step-by-step the actions that are required to install the software on a specific system or computer.

User guides: Describes step-by-step the actions a user shall do to complete specific tasks.

Use cases: Describe step-by-step the actions done by the user, and the feedback the user receives from the interface or system.

Both installation guides and user guides are often written from a second person viewpoint (You shall ...) inviting the reader to step into them, whereas use cases simply lists the events in the order they occur.

Description of a workflow and situation of use

Flexible, progressive, specific and open:

HCS (Human-Centered Stories): These are driven by the emotions and motivations of the characters in them as they try to overcome conflicts or problems, and they may then try to use the interface as a tool to achieve their goals (Strom 2005).

Scenarios: These focus normally on showing how the interface is used in an actual situation of use (a scenario is not necessarily specific, it may also be general and describe a sort of ideal situation of use).

HCS are creative and focused on the users experience, whereas scenarios are descriptions that aims at showing how the different parts of an interface is

used. Both genres may be used to describe real – non-fiction – situations of use or situations of use that may become possible with a new interface.

Description of results of user studies

Structured, descriptive, specific and open:

Personas: This is a composite portrait of the most difficult or demanding user of the system, and at least in principle based on observations of existing users (Cooper 1999). (A persona is not necessarily structured; it may sometimes be flexible and describe the user in a sort of free flowing prose.)

Reports of user studies: These capture what has been observed either during field studies or during laboratory tests. They are normally organized according to topics or problems, so all information relating to a specific problem or topic is placed in the same section. (However, there are a few examples of reports where the results are presented in a sort of free-flowing prose in a more or less haphazard order.)

Whereas personas are structured around the user, his or her background, motivations and tasks, reports of user studies normally focus on what is outside the user: observed tasks, problems and situations of use.

Description of a software design

Structured, generalized and formal:

Pseudocode: Describes the detailed function of a piece of software in something that resembles a high level programming language (Zobel 2004) (Pseudocode is progressive, and it may be flexible).

UML: Describes the detailed data structure of a system or a piece of software (Stevens & Pooley 2000): (In contrast to pseudocode it is static – it describes an unchanging state of the system or software).

Pseudocode is used to give a detailed description of a small piece of software, whereas UML is used to describe the overall structure of a piece of software.

Discussion and conclusion

The different genres used in interaction design and software development was classified according to how the main parts of them were written. The result is a classification where the genres in each group not only share the

same characteristics, but also describe the same type of information, so all genres in one group may serve a similar purpose in a development project.

The classification makes it possible first to choose a genre from a group that fits the given purpose, and then within the group to select the most suitable genre, taking into consideration precisely what it is possible to describe in it.

However, the classification is only a first step. There is a need of more studies of the comparative advantages and disadvantages of the different genres and the best ways to combine them when doing interaction design or software development.

Acknowledgements

Thanks to Hasse Clausen, DIKU, for the discussions that have inspired the writing of this paper.

Literature

Cheney, Theodore A. R. (2001): Writing creative nonfiction, Ten Speed Press, USA

Cooper, Alan (1999): The inmates are running the asylum

Elbow, Peter (1981): Writing with Power, Oxford University Press

Naur, Peter (1995): Logique and the mystique of Logic and Rules, Kluwer Academic Publishers

Naur, Peter (2004): An anatomy of human mental life – Psychology in unideological reconstruction, incorporating The synapse-state theory of mental life, Naur.com publishing

Stevens, Perdita and Rob Pooley (2000): Using UML: Software Engineering with Objects and Components, Pearson Education Limited

Strom, Georg (2005): The reader creates a personal meaning: A comparative study of scenarios and human-centered stories, in People and Computers IX – The Bigger Picture ed. by Tom McEwan, David Benyon & Jan Gulliksen, Springer UK

Strøm, Georg (2005): Noter til Tekstlige beskrivelsesformer – Blok 4, foråret 2005, HCØ tryk, København

White, Hayden (1981): The Value of Narrativity in the Representation of Reality, in On Narrative, ed. by W.J.T. Mitchell, The University of Chicago Press

Zobel. Justin: Writing for computer science, Springer-Verlag London Limited