Context Analysis: Toward Pragmatics of Web Information Systems Design

Hui Ma[1] **Klaus-Dieter Schewe**[1] **Bernhard Thalheim**[2]

[1]Massey University, Department of Information Systems & Information Science Research Centre
Private Bag 11 222, Palmerston North, New Zealand, email: [h.ma|k.d.schewe]@massey.ac.nz

[2]Christian Albrechts University Kiel, Department of Computer Science
Olshausenstr. 40, 24098 Kiel, Germany, email: thalheim@is.informatik.uni-kiel.de

Abstract

On a high level of abstraction a Web Information System (WIS) can be described by a storyboard, which in an abstract way specifies who will be using the system, in which way and for which goals. While syntax and semantics of storyboarding has been well explored, its pragmatics has not. This paper contributes context analysis as a step towards closing this gap. We classify various aspects of contexts related to actors, storyboard, system and time, which make up the context space, then analyse each of these aspects in detail. This is formally support by lifting relations. Finally, we analyse how *contexts* impact on life cases, user models and the storyboard.

1 Introduction

A Web Information System (WIS) is an information system that can be accessed through the world-wide-web. On a high level of abstraction a WIS can be described by a storyboard (Schewe & Thalheim 2005*b*), which in an abstract way specifies who will be using the system, in which way and for which goals. In a nutshell, a *storyboard* consists of three parts:

- a *story space*, which itself consists of a hierarchy of labelled directed graphs called *scenarios*, one of which is the main scenario, whereas the others define the details of *scenes*, i.e. nodes in a higher scenario, and a *plot* that is specified by an assignment-free process, in which the basic actions correspond to the labels of edges in the scenarios,
- a set of *actors*, i.e. abstractions of user groups that are defined by *roles*, which determine obligations and rights, and *user profiles*, which determine user preferences,
- and a set of *tasks* that are associated with *goals* the users may have.

In addition, there are many constraints comprising static, dynamic and deontic constraints for pre- and postconditions, triggering and enabling events, rights and obligations of roles, preference rules for user types, and other dependencies on the plot. Details of storyboarding have been described in (Schewe & Thalheim 2005*b*). An overview of our method for the design of WISs was presented in (Schewe & Thalheim 2005*a*).

While syntax and semantics of storyboarding has been well explored, its pragmatics apart from the use of metaphors (Thalheim & Düsterhöft 2000) has not. Pragmatics is part of semiotics, which is concerned with the relationship between signs, semantic concepts and things of reality. This relationship may be pictured by the so-called semiotics triangle. Main branches of semiotics are *syntactics*, which is concerned with the syntax, i.e. the construction of the language, *semantics*, which is concerned with the interpretation of the words of the language, and *pragmatics*, which is concerned with the current use of utterances by the user and context of words for the user. Pragmatics permits the use of a variety of semantics depending on the user, the application and the technical environment. Most languages defined in Computer Science have a well-defined syntax; some of them possess a well-defined semantics; few of them use pragmatics through which the meaning might be different for different users.

Syntactics is often based on a constructive or generative approach: Given an alphabet and an set of constructors, the language is defined as the set of expressions that can be generated by the constructors. Constructions may be defined on the basis of grammatical rules.

Semantics of generative languages can be either defined by meta-linguistic semantics, e.g. used for defining the semantics of predicate logics, by procedural or referential semantics, e.g. operational semantics used for defining the semantics of programming languages, or by convention-based semantics used in linguistics. Semantics is often defined on the basis of a set of relational structures that correspond to the signature of the language.

Pragmatics has to be distinguished from pragmatism. Pragmatism means a practical approach to problems or affairs. According to Webster (Web 1991) pragmatism is the "balance between principles and practical usage". Here we are concerned with pragmatics, which is based on the behaviour and demands of users, therefore depends on the understanding of users.

The six characteristics of WISs that were discussed in (Schewe & Thalheim 2005*b*) can be mapped to conceptual structures that are used for storyboard specification:

1. We start with the characteristics used for the strategic layer. Main specification elements used are intention and mission. They are mapped to metaphors, general goals, rhetorical figures, and patterns and grids of web pages discussed later.
2. The scenarios reflect the utilisation by actors, for which we envision a number of stories that correspond to real use. These scenarios may be captured through observation of reality. Story spaces and plots are recorded in various level of

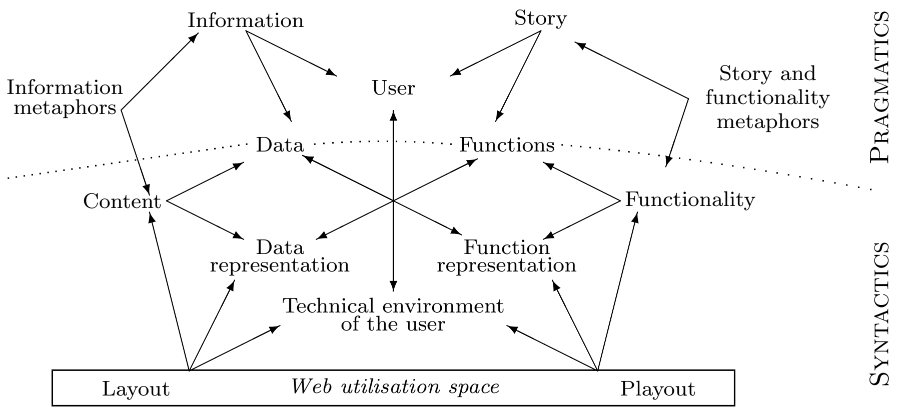

Figure 1.1: The Web Utilization Space Based On the Characteristics of WIS

detail through the methods discussed in (Schewe & Thalheim 2005*b*). The stories are reflected in the storyboard.

3. Content specification is the basis for the media types, i.e. data types and their functions, which will be introduced in part III. It combines data specification with user requirements and is reflected in the content portfolio.

4. Functionality is provided by the media types as required by the storyboard. Typical standard functions are navigation, retrieval (search), support functions, and feedback facilities.

5. Context is based on tasks, history, and environment. We use the specification of context for restructuring and functionality enhancement, which will form the basis of XSL transformations and the onion approach (Binemann-Zdanowicz, Kaschek, Schewe & Thalheim 2004).

6. Presentation depends on the intention, the provider, the technical environment available and the users the WIS is targeting at. Presentation results in the layout and the playout of the WIS. *Layout* requires the development of multimedia presentations for each page. *Playout* additionally requires the development of functionality that supports visits of users depending on the story they are currently following to achieve their goals. Layout and playout integrate the chosen metaphors; they depend on chosen page patterns and grids as well as on quality requirements.

Conceptual structures and their association are depicted in Figure 1.1. We may separate the syntactics and pragmatics layers. Arrows are used for representing part-of- or uses- or relates-associations. For instance, the story is based on the user and the functions. Information metaphors relate content to information.

We use the notions of information and content in a specific manner. *Information* as processed by humans, is carried by *data* that is perceived or noticed, selected and organized by its receiver, because of his subjective human interests, originating from his instincts, feelings, experience, intuition, common sense, values, beliefs, personal knowledge, or wisdom, simultaneously processed by his cognitive and mental processes, and seamlessly integrated in his recallable knowledge. Content is complex and ready-to-use data. Content management systems are information systems that support extraction, storage and delivery of complex data. Content may be enhanced by *concepts* that specify the semantic meaning of content objects and by *topics* that specify the pragmatic understanding of users.

Therefore, information is directed towards pragmatics, whereas content may be considered to highlight the syntactical dimension. If content is enhanced by concepts and topics, then users are able to capture the meaning and the utilisation of the data they receive. In order to ease perception we use *metaphors*. Metaphors may be separated into those that support perception of information and into those that support usage or functionality.

Users are reflected by actors that are abstractions of groups of users. Pragmatics and syntactics share data and functions. The functionality is provided through functions and their representations. The web utilisation space depends on the technical environment of the user. It is specified through the layout and the playout. Layout places content on the basis of a data representation and in dependence of the technical environment. Playout is based on functionality and function representations, and depends on the technical environment.

The *information transfer* from a user A to a user B depends on the users A and B, their abilities to send and to receive the data, to observe the data, and to interpret the data. Let us formalise this process. Let s_X denote the function user by a user X for data extraction, transformation, and sending of data. Let r_X denote the corresponding function for data receival and transformation, and let o_X denote the filtering or observation function. The data currently considered by X is denoted by D_X. Finally, data filtered or observed must be interpreted by the user X and integrated into the knowledge K_X a user X has. Let us denote by i_X the binary function from data and knowledge to knowledge. By default, we extend the function i_X by the time t_{i_X} of the execution of the function.

Thus, the data transfer and information reception (or briefly information transfer) is formally expressed it by

$$I_B \;=\; i_B(o_B(r_B(s_A(D_A))), K_B, t_{i_X})\ .$$

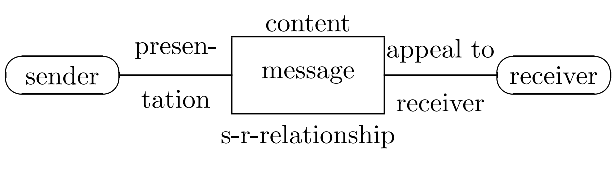

Figure 1.2: Dimensions of understanding messages

In addition, time of sending, receiving, observing, and interpreting can be taken into consideration. In this case we extend the above functions by a time argument. The function s_X is executed at moment t_{s_X}, r_X at t_{r_X}, and o_X at t_{o_X}. We assume $t_{s_A} \leq t_{r_B} \leq t_{o_B} \leq t_{i_B}$ for the time of sending data from A to B. The time of a computation f or data consideration D is denoted by t_f or t_D, respectively. In this extended case the information transfer is formally expressed it by

$$I_B = i_B(o_B(r_B(s_A(D_A, t_{s_A}), t_{r_B}), t_{o_B}), K_B, t_{i_B}) .$$

The notion of information extends the dimensions of understanding of message displayed in Figure 1.2 to a web communication act that considers senders, receivers, their knowledge and experience. Figure 1.3 displays the multi-layering of communication, the influence of explicit knowledge and experience on the interpretation.

The communication act is specified by

- the communication message with the content or content chunk, the characterisation of the relationship between sender and receiver, the data that are transferred and may lead to information or misinformation, and the presentation,
- the sender, the explicit knowledge the sender may use, and the experience the sender has, and
- the receiver, the explicit knowledge the receiver may use, and the experience the receiver has.

In this paper we approach the analysis of WIS usage as the first important part of storyboarding pragmatics. WIS usage analysis consists of three parts:

1. *Life cases* capture observations of user behaviour in reality. They can be used in a pragmatic way to specify the story space. The work on life cases was reported in a previous publication (Schewe & Thalheim 2007*a*).
2. *User models* complement life cases by specifying user and actor profiles, and actor portfolios. The actor portfolios are used to get a better understanding of the tasks associated with the WIS. The work on user models was reported in a previous publication (Schewe & Thalheim 2006).
3. *Contexts* complement life cases and user models by characterising the situation in which a user finds him/herself at a certain time in a particular location. We classify various aspects of contexts related to actors, storyboard, system and time, which make up the context space, then analyse each of these aspects in detail. This is formally support by lifting relations.

After a brief overview of the literature in Section 2 we approach the specification of contexts in Section 3. Formal aspects of contexts are then dealt with in Section 5. In Section 6 we conclude with a brief summary and a discussion of open issues.

2 Related Work

Storyboarding and also the preceding strategic modelling of WIS (Moritz, Schewe & Thalheim 2005) are unique to our approach to WIS modelling. Other approaches to WIS engineering such as the object oriented OOHDM (Güell, Schwabe & Vilain 2000, Rossi, Schwabe & Lyardet 1999, Schwabe & Rossi 1998), WebML (Ceri, Fraternali, Bongio, Brambilla, Comai & Matera 2003), HERA (Houben, Barna, Frasincar & Vdovjak 2003) and variants of UML (Conallen 2003, Lowe, Henderson-Sellers & Gu 2002) concentrate on providing models of content, navigation and interaction by means of extended views, which in our own work is captured by so-called *media types* (Schewe & Thalheim 2005*b*). WSDM (De Troyer & Leune 1998) emphasises the additional need for a mission statement and a high-level description of processes and users in the WIS. Quite often high-level modelling of WISs is subject to UML-based methods, in particular variants of use cases. The rationale underlying our work is that this is far too little to capture strategic and usage issues of WISs.

The integration of goals and soft goals into the information systems development process has been proposed by (Mylopoulos, Fuxman & Giorgini 2000, Giorgini, Mylopoulos, Nicchiarelli & Sebastiani 2002). The detailed distinction of intentions as targets, objects, objectives and aims is based on linguistics (Web 1991). The integration of the temporal dimension is necessary because of the information systems context. The extension by the representational dimensions has been made in order to specify aspects of WISs.

Contextual modelling has found its own research community (Bouquet, Serafini, Brezillon, Benerecetti & Castellani 1999), but very little work has been done to integrate contextual modelling into WIS design. The work in (Mylopoulos & Motschnig-Pitrik 1995) uses contexts as an approach to modularise conceptual models. The most advanced attempt to modelling context is the work on contextual information bases (CIB) (Akaishi, Spyratos & Tanaka 2002, Theodorakis, Analyti, Constantopoulos & Spyratos 1998, Theodorakis, Analyti, Constantopoulos & Spyratos 1999). Roughly speaking, a CIB-context associates objects with a name and an optional reference to another CIB-context. Thus, both the name and the reference depend on the usage history. As shown in (Kaschek, Schewe, Thalheim & Zhang 2004) CIB-contexts in a slightly generalised form can be combined with media types to provide a formal model of usage context. For this, instead of only associating merely a name with a location L a location, i.e. a complex value, is associated with it. Contexts are often defined in a very general and not satisfying manner as a set of associated (data) objects. This definition is used in information retrieval, HCI and other programmer communities. We base our approach on the definition that is commonly accepted in science (Safra, Yeshua & et. al 2003).

3 Context Determination

Taking the commonly accepted meaning a context characterises the situation in which a user finds him/herself at a certain time in a particular location. In this sense context is usually defined only statically referring to the content of a database. Only very few attempts have been made so far to consider context of scenarios or stories.

More generally, we consider context as everything that surrounds a utilisation situation of a WIS by a user and can throw light on its meaning. Therefore, context is characterised by interrelated conditions for the existence and occurrence of the utilisation situation such as the external environment, the internal state, location, time, history, etc. For WISs we need to handle the mental context that is based on the profile of the actor or user, the storyboard context that is based on the story leading to a situation, the data context that is based on the available data, the stakeholder context, and the collaboration context. These different kinds of contexts have an influence on

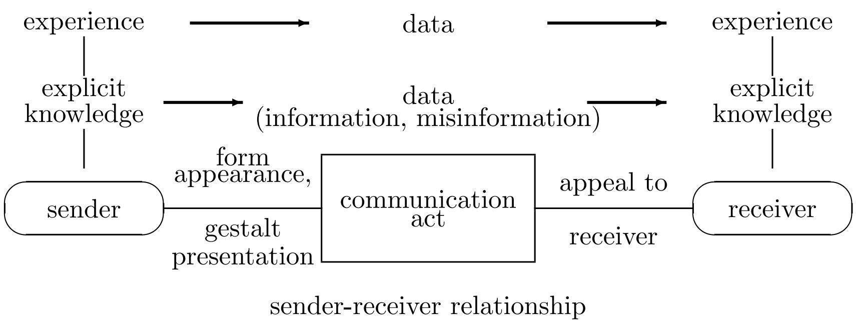

Figure 1.3: Dimensions of the communication act

the development of the storyboard and must thus be considered for the development of the WIS.

Example 3.1 Let us consider a travel information system. It is often desirable to resolve the context of utterances. While booking an airline ticket to *London* the user may be asked for the airport code, for which s/he has a choice between LGW (London Gatwick), LHR (London Heatrow), LON (all London airports in UK), STN (London Stansted), and YXU (London, Ontario, Canada). The context of the travel request can be used to exclude the last option. The context of the airline used so far can be used to exclude two of the others. This context injection is based on the story environment and on content data. ♢

3.1 Context Space

When determining context we already know the major life cases we would like to support, the intentions associated with the WIS, the user and actor characterisation on the basis of profiles and portfolios, and the technical environment we are going to use. These restrictions enable a more refined understanding of context within a WIS.

The work in (Moritz et al. 2005) characterises a WIS by six intertwined dimensions, one of which is context. We now relate context to the other dimensions, i.e. to the intentions, the usage, the content, the functionality, and the presentation. As presentation resides on a lower level of abstraction, it does not have an impact on context. Content and functionality will be used for context refinement, which we address later. So, we first concentrate on intention and usage. The user model, the specified set of life cases, and the intention can be used for a disambiguation of the meaning and an injection of context. In doing so we distinguish the following facets of context:

Actor context: The WIS is used by actors for a number of tasks in a variety of involvements and well understood collaboration. These actors impose their quality requirements on the WIS usage as described by their security and privacy profiles. They need additional auxiliary data and auxiliary functions. The variability of use is restricted by the actor's context, which covers the actor's specific tasks and specific data and function demand, and by chosen involvement, while the profile of actors imposes exceptions. The involvement and collaboration of actors is based on assumptions of social behaviour and restrictions due to organisational decisions. These assumptions and restrictions are components of the actor's context.

Storyboard context: The meaning of content and functionality to users depends on the stories, which are based on scenarios that reflect life cases and the portfolios of users or actors. According to the profile of these users a number of quality requirements such as privacy, security and availability must be satisfied. The actor's scenario context describes what the actor needs to understand in order to efficiently and effectively solve his/her tasks in the actual portfolio. The actor's determine the policy for following particular stories.

System context: The WIS is developed to support a number of intentions. The purposes and intents lead to a number of decisions on the WIS architecture, the technical environment, and the implementation. The WIS architecture has an impact on its utilisation, which often is only implicit and thus leads to not understandable systems behaviour. The technical environment restricts the user due to restrictions imposed by server, channel and client properties. Adaptation to the current environment is defined as context adaptation to the current channel, to the client infrastructure and to the server load. At the same time a number of legal decisions based on regulations, laws and business rules have been incorporated into the WIS.

Temporal context: The utilisation of a scene by an actor depends on his/her history of utilisation. Actors may interrupt and resume their activities at any moment of time. As they may not be interested in repeating all previous actions they have alrady successfully completed, the temporal context must be taken into account. Due to availability of content and functionality the current utilisation may lead to a different story within the same scenario.

We will discuss these various facets of context in more detail later in this section. This entire information forms the *context space*, which brings together the storyboard specification and the contextual information. Typical questions that are answered on the basis of the context space are:

- What content is required by the context space?
- What functionality is required by the context space?
- What has to be changed for the life cases, the storyboard, etc., if context is considered?

As outlined above the context space is determined by the actors, the scenarios, the WIS itself, and the time. It leads to a specialisation of the content, structuring and functionality of the scenes.

Context is associated with *desirable properties* of the WIS such as quality criteria and security and privacy requirements. Quality criteria such as suitability for the users or learnability provide obligations for the WIS development process. Though these criteria are rather fuzzy, they lead directly to a number of implementation obligations that must be fulfilled at later stages, i.e. within the development on the implementation layer.

For instance, learnability means comprehensibility, i.e. the WIS must be easy to use, remember, capture and forecast. This requires clarity of the visual representation, predictability, directness and intuitiveness. These properties allow the user to concentrate on the tasks. The workflows and the discourse structure correspond to the expectations of the users and do not lead to surprising situations. They can be based on metaphors and motives taken from the application domain. In the same way other quality criteria can also be mapped to development obligations.

Other properties that may be associated with context refer to the potential utilisation for other tasks outside the scope of the storyboard. In this case we do not integrate the additional tasks into the storyboard, but instead support these tasks, if this in accordance with our intentions. For instance, we might expect further visits targeting at core concerns of the WIS.

Example 3.2 Sometimes customers may want to use a WIS for a purpose that does not meet the system's mission statement. For example, a customer may use a banking WIS to learn about the loan business, or a bookshop WIS to learn English. Clearly, the larger the gap between the actual customer's intention and the system's mission statement is, the higher the expected costs will be for supporting such customers. If it can be expected that some customers will interact with the WIS in a 'non-standard' way, a decision has to be made whether to support such intentions or not. This implies a modification of the anticipated information space. It shows that our focus on a business model for context modelling is not always a severe restriction. ◊

3.2 Additional Aspects

We may consider three additional context facets:

Provider context: Providers are characterised by their mission, intentions, and specific policies. Additionally, terms of business may be added. Vendors need to understand how to run the WIS economically. Typical parts of this context are intentions of the provider, themes of the website, mission or corporate identity of the site, and occasion and purpose of the visits of actors. Thus, providers may require additional content and functionality due to their mission and policy. They may apply their terms of business and may require a specific layout and playout.

Based on this information, the WIS is extended by provider-specific content and functionality. The storyboard may be altered according to the intentions of the provider, and life cases may be extended or partially supported. Provider-based changes to portfolios are typical for WISs in e-government and e-business applications.

Developer context: The WIS implementation depends on the capability of the developer. Typically we need to take into account the potential environment, e.g. hard- and software, communication channels, the information systems that are to be incorporated, especially the associated databases, and the programming environment developers use.

Organisational and social context: The organisation of task solutions is often already predetermined by the application domain. It follows organisational structures within the institutions involved. We captured a part of these structures already on the basis of the portfolio and modelled it by collaboration. The other pars form the organisational context. Collaboration of partners consists of communication, coordination, and cooperation. Cooperation is based on cooperativity, i.e. the disposition to act in a way that is best helpful for the collaboration partners, taking their intentions, tasks, interests and abilities into account. At the same time, collaboration is established in order to achieve a common goal. Actors choose their actions and organise them such that their chances of success are optimised with respect to the portfolio they are engaged in. Additionally, the social context may be taken into account, which consists of interactive and reactive pressures. Typical social enhancements are socially indicated situations such as welcome greetings, thanking, apologising, and farewell greetings.

4 Details of Context Specifications

Let us next take a deeper look into the facets of the context space, i.e. examining actor, storyboard, system and temporal context in more detail.

4.1 Actor Context

The context of an actor is based on his/her intentions. According to the actor's profile s/he needs support to fulfil the expectations with respect to the quality of information and work. The social and intellectual interests of the actor may also be part of the actor's context. The actor's profile may be used for a refinement of the actor's context leading to the following four specific kinds of context:

Actor projection context: Actors may act on their expectations. In this case, they intensionally drop portions of content or functionality and project the current content and functionality to the "normal" case. This projection leads to an implicit context. For instance, within a travel scenario actors are expected to behave like travellers. Another kind of projection is parameter suppression, in which case content or functionality may be dropped or is not noticed whenever it becomes partially irrelevant.

Actor approximation context: Often actors need first a condensed or approximated information that may be refined later. Typically such approximations are attribute value approximations or structural approximations. For instance, the former ones may allow the WIS to provide first an approximate value for the orientation of the user. A common misuse of approximation is pricing by "starting from". Structural approximation permits the use of the same symbol for the original object and an abstraction, hence enables the usage of simpler representations.

Actor ambiguity context: Sometimes the reference of a symbol can be unambiguous within a narrow scope, in which certain limitations apply, but ambiguous in a larger scope without the limitations. A typical unambiguous symbol is the 'next' button in case the next scene lies within the expectations of the actor. Another use of ambiguity can be made by choosing less expressive textual representations. For instance, in a loan application there is no need to clarify that the word 'bank' denotes a financial institution.

Actor mental context: The mental context captures attitudes and knowledge of actors or other kinds of alternative states of affairs such as fiction and user expectations. This context is described in terms of provenance, i.e. relating to real life cases or to expected life cases. Expectations of actors or users can be combined with other more general requirements. The knowledge of the mental context will remain highly incomplete. However, it provides a handle for incorporating users' and actors' expectations.

The actor context is intellectual as well as existential. It contributes to enabling the scenarios and the corresponding stories under consideration. The intellectual part is based on the profile of the actor, on habits, traditions, knowledge, experience, etc. It may be also based on the quality requirements an actor is imposing. The actor context restricts the users that might use the WIS, the way the system will be used, and the portfolios. It is based on the intentions for using the system and the portfolio of the actor, e.g. tasks, involvement, and the collaborations the actor is involved in. The existential part is also related to the portfolio under consideration. It is related to the data and functions currently available or provided, and the technical environment.

The specification of this specific actor context becomes necessary whenever we want to support the work of actors that is close to human communication. Human communication exploits the context often to an extreme degree, leaving many things implicit. We do not need a complete decontextualisation as long as the actor can interpret the content and functionality that is provided by the WIS. Contexts provide a mechanism by which we can use the simplest presentation, content and functionality, i.e. the ones that makes the fewest distinctions most of the time while transcending to more expressive presentation, content and functionality only when needed.

Due to these contextual abilities we may restrict presentation, content and functionality to those features that are absolutely necessary. These restrictions may result in presentation principles such as sparse utilisation of additional and not directly necessary content or functionality or economy in utilisation of colours, multimedia objects, and texture.

EXAMPLE 4.1 Let us use relocation as an example for an illustration of this principles. An issuer of the relocation life case expects that his personal and identification data are already sufficient for providing him/her all necessary details. So, the context in which the issuer reacts is based on projection and ambiguity context. If we use the information the passport office provides as public information for the city office, then we can adapt the life case directly to the current one. At the same time, the visit of the issuer might be not the first such in his/her life. So, we can now use the information on previous life cases for scaling the life case to the expectations the issuer has.

The adaptation requires some background knowledge on the handling of life cases in other cities, previous visits, and the profile of the issuer. We may then use a number of questions to figure out, which further adaption or refinement of the life case is applicable. Since some data on the issuer cannot be stored in the system due to regulations and laws we need to repeatedly obtain these data. So, the data we need to capture within the life case are extended by data we need for figuring out which specific life case is under consideration. At the same time we may use this context information for adapting functionality that is provided. ◊

This specific actor context is combined with the portfolio restrictions. Actors with a non-deterministic behaviour do not use high ambiguity or deep projection. At the same time, their mental context and their approximation context must be rather sophisticated. Actors acting more on intention intensively use all four kinds of actor contexts. Task-oriented and reactive behaviour requires support for mental context. Actors acting in collaborations need additional support for their common disambiguation. If actors do not complete their tasks within one session, they need a well-prepared projection context for the case that they resume their tasks. We shall later map these requirements to adaptation rules and control rules for adaptation.

4.2 Storyboard Context

Context has also a storyboard dimension. The actor's context must be combined with the storyboard, life case and portfolio contexts. The latter two selectively condition the situational interest of the actor and the relevance of the current scene for the actor. Based on the relevance we may identify and use properly all the content that should exert the evolution of the current story. We may now use this information for extracting whether a sequence rule, i.e. a rule of the form $s_1 \Rightarrow s_2$ requiring that a visit of scene s_1 should be followed by a visit of scene s_2 can be applied to the current system usage. The rule may hold in general, but is considered to be not applicable if the existence of process p_1 leading to scene s_1 does not have a bearing on the existence of process p_2 leading to scene s_2. Therefore, the incorporation of context and the derivation of relevance has mainly to do with selecting the best story for the user and is thus used for the adaptation of content and functionality.

The storyboard context can be used for deriving the most appropriate content. We aim at delivering the right data to the right actor with the right tools and scope at the right time. As the storyboard context provides a good source for adaptation of content and functionality to the current stage of the scenario we collect context within the storyboard and add this context information to the context space. This context allows a treatment of the expectations of the actor. Therefore, each scene in a scenario is provided with a *pre-scene context*, *scene context*, and a *post-scene context*.

- The pre-scene context consists of all content that has already been delivered to the actor before the appearance at the actual scene. This information can be used to reduce content delivery for scenes. At the same time, this content can be stored in condensed form and made available to the actor when needed, i.e. the actor can revisit the old content whenever this seems to be necessary. Classical browsers only provide a strictly sequential 'back' button for this kind of history management. The pre-context of a scene thus contains all valuable content that is collected during a story, and guarantees the availability of this content when needed.
- The post-scene context consists of a potential playout of scenes that can be entered after the current scene. If an actor needs some information on the next actions, then this context information can be used. This information is valuable

for those actors who intend to drop out of the system. It is also a part of the help information. The post-scene context can be enriched by meta-data describing the content that is provided in the next steps or the data to be produced by the actor. In this case, an intelligent interface may forecast the information need for further steps of the storyboard.

- Each scene may also be enriched by superimposed meta-data on the scene, which include everything that could be referenced within the expected consumed and produced data. Typical such references are collaborating actors, retrieval or update data of the current content, and details taken from the log of the current story. Finally, the scene context may include administrative data such as identification of content currently under consideration.

  Scene context is enhanced by generic scene information, which can be based on intentions of the WIS. For instance, adverts may be attached to each of the scenes. Default information serves as an exception handling for scenes. If content or functions are currently not available, then default data are provided.

4.3 System Context

The system context is determined by the content and the functions that are provided by the web information system. It consists of at least the following four parts:

Source and acquisition: Source and acquisition is an orthogonal dimension of the WIS. A WIS is supported by media objects that belong to media types as we will explore in detail in part III. In a nutshell, a media object is defined by an extended view on some underlying database, which can then serve for provision of content and functionality of an elementary scene. The databases used for the generation of content form the context of the scenario. We may associate with each scenario the subschemata of these databases that are used for generation of consumed data or for integration of data produced by actors in the scenario.

Associated content: The data that are used for consumption and production of information do not exist in isolation. They are usually associated with other data on the basis of integrity constraints or existence constraints, in particular existence constraints are often not explicitly represented as such, but are embedded into the database schemata used. For instance, we usually associate with objects collected in relationship classes those objects collected in the component classes on which they are based. In this case, we assume that objects in component classes of the relationship type co-exist with objects in the relationship class. We need to consider the environment of content that is currently under consideration together with the data that are associated with this content.

Supported functionality: Functions supporting the actions in scenarios are provided by the WIS. These functions have their own control environment. Typical such control mechanisms are logging, concurrency control, and recovery management.

Security: Security concepts describe *encipherment* and *encryption* (keys, authentication, signatures, notarisation, routing control, access control, data integrity, and traffic padding) for data exchange. They can be enriched by other system-supported security concepts and enhanced by privacy concepts.

4.4 Temporal Context

The temporal context appears in a number of variants, e.g. storage time, validity time, display time, user-defined time, transaction time, etc. The temporal context is applicable in a number of combinations. Sometimes it is necessary to use all of them, but often it is often observed that only one variant of this context is necessary.

Versions show the life cycle of the objects under consideration. As scenarios will have their own life cycle we cannot assume that database changes are directly enforced on websites. Moreover, it may be useful to provide the old content as long as an actor continues with the same story. Versions can often be systematically structured by *database system phases*:

- The *initialization phase* permits the development of objects storing initial information. Integrity constraints are applicable in a limited form.

- The *production phase* is the central phase, which consists of runtime querying, modification, transaction management, etc.

- The *maintenance phase* is used in productive database applications for clarification of soft constraints, maintenance of constraints that have been cut out from runtime maintenance, and changing the structuring and functionality of the entire database system. Maintenance phases are used in data warehouse applications for recharging the data warehouse with actual information.

- The *archiving phase* is used for archiving the content of a database in a form that data relevant for historical information can be easily retrieved. No data modification is permitted; the only modification operation is to load new changes to the archive.

4.5 Context Templates

We may now combine this context information using the following semi-formal template:

`Context:`	⟨context name⟩
`Extension of:`	⟨General context⟩
`Actor context:`	⟨general description⟩
`Projection context:`	⟨expectations⟩
`Approximation context:`	⟨condensations and abstractions⟩
`Ambiguity context:`	⟨scope⟩
`Mental state context:`	⟨general description⟩
`Characterisation:`	⟨general description⟩
`Storyboard context:`	⟨general description⟩
`Pre-scene context:`	⟨history of usage⟩
`Post-scene context:`	⟨potential continuation⟩
`Scene context:`	⟨superimposed meta-data⟩
`WIS context:`	⟨general description⟩
`Source and acquisition:`	⟨system environment⟩
`Associated content:`	⟨content environment⟩
`Supported functionality:`	⟨function environment⟩
`Security:`	⟨required security functionality⟩
`Temporal context:`	⟨general description⟩
`Versioning:`	⟨general description⟩
`Development phase:`	⟨general description⟩
`Provider context:`	⟨general description⟩
`Developer context:`	⟨general description⟩
`Organisational and social context:`	⟨general description⟩
`Based On:`	⟨life cases, portfolio⟩
`Based On:`	⟨scenarios⟩
`Based On:`	⟨ general tasks, audience⟩
`Based On:`	⟨mission, goals⟩

5 Formal Aspects of Context Modelling

Context evolves for actors, scenarios, systems, and over time. We model the relation between different contexts by *lifting relations*. Properties that are valid for a certain context may be lifted to another context. This transfer can be based on local model semantics.

For this recall that a context is determined by actor, storyboard, system and temporal contexts. So let $\mathcal{A}$ denote the set of actors, $\mathcal{S}$ the set of scenarios, $\mathcal{W}$ the set of system characteristics, and $\mathcal{T}$ the set of time units. Then we can take a subset $\mathcal{C} \subseteq \mathcal{A} \times \mathcal{S} \times \mathcal{W} \times \mathcal{T}$ to represent a set of contexts. Furthermore, we use a family of contexts $\{C_i \in \mathcal{C} | i \in I\}$ and a family of statement sets (or theories) $\{\mathbb{T}_i | i \in I\}$ that are associated with these contexts. Of course, the theory $\mathbb{T}_i$ describes the properties of the context C_i.

5.1 Lifting Relations

On these grounds we may use local models $M_{i,j}$ for each of these statement sets assuming that the models we consider are enumerated by the second index. More precisely, the models $M_{i,j}$ determine the meaning of content drawn from a language $\mathcal{L}$ for describing content in view of context C_i. That is, we use a partial mapping $\Psi : \mathcal{L} \times \mathcal{C} \to \mathcal{M}$, where $\mathcal{M}$ denotes a set of pre-determined meanings for content in $\mathcal{L}$.

We may now distinguish the formula α occurring context C_i from the same formula occurring in another context by considering the context index i, i.e. we consider pairs (α, i). Lifting relations can be modelled by rules of the form

$$\frac{(\alpha_1, i_1) \dots (\alpha_n, i_n)}{(\alpha, i)} \varphi$$

stating that the formulae $(\alpha_1, i_1) \dots (\alpha_n, i_n)$ can be lifted to (α, i) under the side condition φ. In addition, a compatibility relation among local models is introduced similar to logics that capture possible world semantics. This compatibility relation is used for entailment and satisfiability. This approach allows us to reason locally and then to transfer the knowledge we gained to other contexts.

Based on this coarse clarification of basic notation we develop a number of facilities and extend the specification of the WIS:

Context space: The *content context space* is defined on the basis of the content $\mathcal{C}$, scenarios $\mathcal{S}$ and actors $\mathcal{A}$. In Example 3.1 we could use information on the travel and on the airline to exclude options that seem to be less likely. The content context space of a WIS for a given content-meaning pair $c, m)$ consists of precisely those contexts, under which the particular content will have that particular meaning, i.e.

$$\mathfrak{C}(c, m) = \{(a, s, w, t) \in \mathcal{C} \mid \Psi(c, (a, s, w, t)) = m\}.$$

Adaptation of content, functionality, and scenarios to the context that is currently available is based on *context infusion*. Applying transformation rules we change content, functions, and the presentation. Therefore, we use a context specification for the development of enforcement rules. These rules may restrict scenarios to more specific ones, extend or shrink content, and extend or remove functions.

Life case extension and specialisation: The general life case specification can often be specialised, if context is explicitly injected. We need both the more general life cases and the contextualised ones. Whenever the WIS is revised or extended, we can return to more general life cases and generate another contextualisation. Typical specialisations concern changes in the life case flow. We may specialise the data consumed by an actor in dependence of the actor's context. If we know that actors need special auxiliary information or conversely actors became more knowledgeable during the utilisation of the WIS, then we may adapt the data provided for consumption. At the same time, we can specialise the figures according to the given context. In the same way spatial and temporal information provide a basis for refinement of life cases.

Life cases may be extended to requirements that were collected in the context space. The content context may require a more elaborated content to be provided. The supported functionality may require additional functions, content, or a specific presentation. Intentions may be more specific under consideration of context. For instance, if we want to support a certain usage of a WIS that was not originally intended but became important in order to maintain frequent visits, then the original life case is extended by those associated life cases.

Development of a context manager: Context is also bound to scenes and thus evolves within a story. We may expect that content enhances context. For this reason, we introduced the prescene context. Therefore, a subsystem that manages the context is needed. This context manager uses the lifting rules introduced above for transferring context to context for scenes, collaborating actors, and the WIS as such. The system also supports the rule-based development of logics over time. We cannot require that the rule system is complete, but it must be consistent. A useful property is commutativity, i.e. the results of firing rules does not depend on their order. The context management system enhances the dialogue management system by adapting and specialising the presentation and injecting context into it.

Example 5.1 A typical context extension to functionality is associated with the problem to avoid that users trap into losing-track situations. Such situationd can be detected based on the user's behaviour, e.g. invoking the help function repeatedly on similar topics, repeatedly positioning on particular locations and performing similar operations on similar data, excessively navigating through information space without invoking any reasonable functionality, looking repeatedly for FAQs on similar topics, attempting to enter a discussion forum, and sending email to the site administrator.

User aid that can be provided for losing-track situations is giving access to a thesaurus of the subsystem the user is accessing. Furthermore, the respective business model may be exposed to the user together with an explanation that is adapted to a particular user type. Similarly, access to a FAQ list suitable for the user and the accessed subsystem may be given. Furthermore, improved search facilities and examples targeting at the subsystem accessed may be provided. ◊

5.2 Adaptivity

The idea of adaptivity is to equip the system with enough additional information and rules that would render it possible to engender the right content and functionality for the current situation. That is, the system is supposed to act according to the dictum 'you take care of the specification, and the system will take care of itself and adapt to the current use'.

Two content objects c_1, c_2 are *synonymous* in the context $C_i \in \mathcal{C}$ iff $\psi(c_1, C_i) = \psi(c_2, C_i)$. They are *totally synonymous* iff $\psi(c_1, C_i) = \psi(c_2, C_i)$ holds for all contexts $C_i \in \mathcal{C}$. They are *epistemically synonymous* within a scenario s for an actor a iff $\psi(c_1, C_i) = \psi(c_2, C_i)$ holds for all contexts $C_i \in \mathcal{C}$ associated with a and s.

Applications often require adaptation of processing context, e.g. to

- actual environments such as client, server, and current communication channel,
- user rights, roles, obligations, and prohibitions,
- content required for the portfolio of the current user,
- the actual user with his/her preferences,
- the level of task completion depending on the user, and
- the user's completion history.

Consider for instance e-learning or e-government websites discussed in (Moritz et al. 2005) and (Schewe, Thalheim, Binemann-Zdanowicz, Kaschek, Kuss & Tschiedel 2005). Citizens may apply for a primary place of residence. In this case, their passport must be changed; otherwise, no change is required. Citizens with school-age children may have to complete additional documents. Completed documents may be decomposed into a suite of documents due to legal restrictions, e.g. by a data protection act requiring that data for city officials and service offices such as the unemployment agency must be separated.

Depending on the role of users, story completion may be scheduled sequentially for some users or in parallel for others. For instance, clerks in a city office may consider documents in parallel, while citizens complete their documents in a sequential mode.

EXAMPLE 5.2 Adaptivity may be required at runtime. For instance, people with foreign citizenship may be required to apply for a residence permit. Users may require a varying support depending on the environment that is used for the completion of documents. Users should be supported whenever they are interrupted during task completion. $\diamond$

These requirements lead directly to the requirement to develop a facility for *mutable, adaptable scenarios* for different users, portfolios, and contexts. We shall return to this requirement after introducing templates in the next section. It is our target to develop generic scenarios that can be refined to scenarios by injecting context. This approach is more widely used for WISs than one would expect. For instance, almost all information sites of cities and regions provide a very similar hotel or event search. The reason is not the existence of a development monopoly but rather the evolution of these search facilities to semi-standards. These standards are not officially agreed, but have been formed by copying successful solutions.

6 Conclusion

In this paper we approached the pragmatics of Web Information Systems (WIS) design focusing on the method of storyboarding that is an integral part of the codesign approach to WIS design (Schewe & Thalheim 2005*a*, Schewe & Thalheim 2005*b*). A storyboard specifies in an abstract way who will be using the WIS, in which way, and for which goals. Thus, the specification of a storyboard captures the navigation paths, i.e. the stories through the "scenes" of the WIS, the action scheme accociated with the stories, the actors appearing in the scenes, and the tasks the actors accomplish. In addition, there are various static, dynamic and deontic constraints governing the storyboard.

While syntax and semantics of storyboarding have been well explored, its pragmatics has not. While many methods for WIS design emphasise content modelling, we start from the very fundamental observation grounded in semiotics that content refers to a syntactic dimension, whereas a pragmatic dimension requires dealing with information. This led to the objective to investigate in depth intentions associated with a WIS. The facets of intention arising from this form the basis for our technical development in this paper dealing with life cases, user models, and contexts.

Life cases capture observations in reality, which by means of abstraction can be used to derive scenarios for the storyboard. Integrating these scenarios provides a method for storyboarding. User models are by user and actor profiles, and actor portfolios. The latter ones provide a better understanding of the tasks associated with the WIS. Contexts can be classified according to how they impact on the life cases, the user models, and the storyboard extracted from them.

This work on pragmatics of storyboarding contributes to closing a gap in the codesign methodology for WIS design. It links the formalism of storyboarding to the systems requirements, and provides guidelines and means to derive the complex storyboards from informal ideas about a WIS without any technical bias. So, one one hand, this work on pragmatics is a decisive part of the methodology, which does not just consist of a collection of formally integrated models, but also has to state how to use them. It would be rather difficult mapping life cases or user models directly to a conceptual model of a WIS, which resides on a much lower level of abstraction as the storyboard. So on the other hand this work emphasises the need for storyboarding as the decisive tool for high-level WIS engineering. As shown in (Schewe & Thalheim 2007*b*) this is also the basis for high-level reasoning about WISs addressing such important issues as personalisation of functionality.

Despite the high relevance of pragmatics for the completeness of storyboarding and the codesign methodology as a whole, the work reported in this paper is only part of the story, as it only addresses the context analysis. Together lith life cases (Schewe & Thalheim 2007*a*) and user models (Schewe & Thalheim 2006) they capture usage analysis of WISs, which still does not completely exhaust the problem area associated with pragmatics of storyboarding. We are in the process of writing up a second part of storyboarding pragmatics dealing with WIS portfolios, which combines content and utilisation portfolios that give rise to content and functionality chunks. The content portfolio is used for collecting information requirements. It is based on information needs and demands, and links the storyboard to the lower-level conceptual model of the WIS consisting of a collection of media types. The utilisation portfolio is used for collecting functionality requirements. It describes intentions of users, specific needs and their context.

In addition to this follow-on part on storyborading pragmatics we are also working on the pragmatism of storyboarding, storyboard refinements, and quality evaluation. All this together plus the ongoing research on logical grounds of storyboarding and their exploitation for reasoning and verification will complete our research on high-level WIS design within the codesign framework.

References

Akaishi, M., Spyratos, N. & Tanaka, Y. (2002), A component-based application framework for context-driven information access, *in* H. Kangassalo et al., eds, 'Information Modelling and Knowledge Bases XIII', IOS Press, pp. 254–265.

Binemann-Zdanowicz, A., Kaschek, R., Schewe, K.-D. & Thalheim, B. (2004), Context-aware web information systems, *in* 'APCCM', pp. 37–48.

Bouquet, P., Serafini, L., Brezillon, P., Benerecetti, M. & Castellani, F., eds (1999), *Modeling and Using Context – Context'99*, Vol. 1688 of *LNAI*, Springer-Verlag.

Ceri, S., Fraternali, P., Bongio, A., Brambilla, M., Comai, S. & Matera, M. (2003), *Designing Data-Intensive Web Applications*, Morgan Kaufmann, San Francisco.

Conallen, J. (2003), *Building Web Applications with UML*, Addison-Wesley, Boston.

De Troyer, O. & Leune, C. (1998), WSDM: A user-centered design method for web sites, *in* 'Computer Networks and ISDN Systems – Proceedings of the 7th International WWW Conference', Elsevier, pp. 85–94.

Giorgini, P., Mylopoulos, J., Nicchiarelli, E. & Sebastiani, R. (2002), Reasoning with goal models, *in* 'ER', pp. 167–181.

Güell, N., Schwabe, D. & Vilain, P. (2000), Modeling interactions and navigation in web applications, *in* S. W. Liddle, H. C. Mayr & B. Thalheim, eds, 'Conceptual Modeling for E-Business and the Web', Vol. 1921 of *LNCS*, Springer-Verlag, pp. 115–127.

Houben, G.-J., Barna, P., Frasincar, F. & Vdovjak, R. (2003), HERA: Development of semantic web information systems, *in* 'Third International Conference on Web Engineering – ICWE 2003', Vol. 2722 of *LNCS*, Springer-Verlag, pp. 529–538.

Kaschek, R., Schewe, K.-D., Thalheim, B. & Zhang, L. (2004), Integrating context in conceptual modelling for web information systems, *in* C. Bussler, D. Fensel, M. E. Orlowska & J. Yang, eds, 'Web Services, E-Business, and the Semantic Web', Vol. 3095 of *LNCS*, Springer-Verlag, pp. 77–88.

Lowe, D., Henderson-Sellers, B. & Gu, A. (2002), Web extensions to UML: Using the MVC triad, *in* S. Spaccapietra, S. T. March & Y. Kambayashi, eds, 'Conceptual Modeling – ER 2002', Vol. 2503 of *LNCS*, Springer-Verlag, pp. 105–119.

Moritz, T., Schewe, K.-D. & Thalheim, B. (2005), 'Strategic modelling of web information systems', *International Journal on Web Information Systems* **1**(4), 77–94.

Mylopoulos, J., Fuxman, A. & Giorgini, P. (2000), From entities and relationships to social actors and dependencies, *in* 'Conceptual Modeling - ER 2000', Springer-Verlag, Berlin, pp. 27–36.

Mylopoulos, J. & Motschnig-Pitrik, R. (1995), Partioning information bases with contexts, *in* 'Proc. CoopIS '95', pp. 44–55.

Rossi, G., Schwabe, D. & Lyardet, F. (1999), Web application models are more than conceptual models, *in* P. Chen et al., eds, 'Advances in Conceptual Modeling', Vol. 1727 of *LNCS*, Springer-Verlag, Berlin, pp. 239–252.

Safra, J., Yeshua, I. & et. al (2003), *Encyclopedia Britannica*, Merriam-Webster.

Schewe, K.-D. & Thalheim, B. (2005*a*), 'The codesign approach to web information systems development', *International Journal on Web Information Systems* **1**(1), 5–14.

Schewe, K.-D. & Thalheim, B. (2005*b*), 'Conceptual modelling of web information systems', *Data and Knowledge Engineering* **54**(2), 147–188.

Schewe, K.-D. & Thalheim, B. (2006), User models: A contribution to pragmatics of web information systems design, *in* K. Aberer, Z. Peng & E. Rundensteiner, eds, 'Web Information Systems – Proceedings WISE 2006', Vol. 4255 of *LNCS*, Springer-Verlag, pp. 512–523.

Schewe, K.-D. & Thalheim, B. (2007*a*), Life cases: An approach to address pragmatics in the design of web information systems, *in* 'Proceedings WebIST'07'.

Schewe, K.-D. & Thalheim, B. (2007*b*), 'Personalisation of web information systems - a term rewriting approach', *Data and Knowledge Engineering* **62**(1), 101–117.

Schewe, K.-D., Thalheim, B., Binemann-Zdanowicz, A., Kaschek, R., Kuss, T. & Tschiedel, B. (2005), 'A conceptual view of electronic learning systems', *Education and Information Technologies* **10**(1-2), 83–110.

Schwabe, D. & Rossi, G. (1998), 'An object oriented approach to web-based application design', *TAPOS* **4**(4), 207–225.

Thalheim, B. & Düsterhöft, A. (2000), 'The use of metaphorical structures for internet sites', *Data & Knowledge Engineering* **35**, 161–180.

Theodorakis, M., Analyti, A., Constantopoulos, P. & Spyratos, N. (1998), Context in information bases, *in* 'Proc. CoopIS '98', pp. 260–270.

Theodorakis, M., Analyti, A., Constantopoulos, P. & Spyratos, N. (1999), Contextualization as an abstraction mechanism for conceptual modelling, *in* 'Conceptual Modeling – Proc. ER'99', Vol. 1728 of *LNCS*, Springer-Verlag, pp. 475–489.

Web (1991), 'Webster's ninth new collegiate dictionary'.