INSTITUT FÜR INFORMATIK

Intention-Driven Screenography

René Noack, Bernhard Thalheim

Bericht Nr. 0712
December 2007

CHRISTIAN-ALBRECHTS-UNIVERSITÄT

KIEL

Institut für Informatik der
Christian-Albrechts-Universität zu Kiel
Olshausenstr. 40
D – 24098 Kiel

Intention-Driven Screenography

René Noack, Bernhard Thalheim

Bericht Nr. 0712
December 2007

e-mail: {noack|thalheim}@is.informatik.uni-kiel.de

Intention-Driven Screenography

Thomas Moritz[1], René Noack[2],
Klaus-Dieter Schewe[3], Bernhard Thalheim[2]

[1]Brandenburg University of Technology, Institute of Computer Science
Cottbus, Germany moritz.th@t-online.de
[2]Christian Albrechts University Kiel, Department of Computer Science
Kiel, Germany noack|thalheim@is.informatik.uni-kiel.de
[3]Massey University, Information Science Research Centre
Palmerston North, New Zealand k.d.schewe@massey.ac.nz

Abstract: The visual design development of Web Information Systems is a complex task. At present, the process is mainly based on experiences and seems to be an immovable part of art. Typically, occurs a late consideration of graphical issues that results in inflexibility and cause problems for extension and change management. Database and software systems are mainly based on development phases such as requirement acquisition and elicitation and conceptual modelling. Moreover, users, their preferences and portfolio are taken into consideration. We show in this preprint that these approaches can be generalised to website presentation. We use methods developed for programming in the large, e.g. patterns. We can map patterns to conceptualisations of web page layout, i.e. grids. Patterns shall help us to reuse concepts. This paper introduces the concept of pattern and clarifies their structure and task for the whole development. Because the WIS development process is based on six dimensions, we initially introduce development dimensions and show the seamless integration of the pattern-based approach. We call the art of website layout screenography. Screenography extends web application engineering by scenographic and dramaturgic aspects and intends to support the interaction between system and user. Screenography aims at an individualised, decorated playout in consideration of intention, user profiles and portfolios, provider aims, context, equipment, functionality and the storyline progress. The users orientation of WIS requires the deep integration of user concerns, tasks and expectations into screenography. Therefore, this paper develops concepts of intention-driven screenography.

1 Introduction

In general, we are able to perceive sites as good sites and bad sites in terms of impression, composition, usability, and utility, in dependency on our subjective preferences and needs. In general there are no methodologies to define overall good sites, because the perception depends, for example, on users recognition resp. their profile. Nevertheless, we sense well designed sites as well structured and

well orchestrated, which isn't a result of coincidence. At present, the development process is often based on experiences of a developer, because the absence of global layout development concepts. The reasons for development decisions are as far as possible unknown and seems to be an immovable part of art. Because we try to turn the development process from an art into a craft, we have to detect the basic principles of good sites, as far as possible. Therefore we analyse existing sites, searching for multiple used patterns, and try to understand their applicability. Moreover we have to consider other approaches like task models, e.g. CTT [Pat99], as well as website description languages, e.g. SiteLang [TD01]. The main objective of this direction is the development process generalisation. We aim in handling development decisions to derive an adequate layout from generic descriptions. Therefore we have to differ WE techniques, e.g. XML or techniques for content management, from HCI techniques, which are more interaction resp. presentation oriented. To close the gap between the WE and HCI approach is an aim for future research.

However, we missing a full story integration at present, and the important part of user adaptation exists only rudimentary in most cases. Another deficiency is the development from the scratch, piece by piece each time. In general the recycling of parts is used on a very low level, because the layout is often examined as a big black box developed by a designer. Conceptual approaches only rudimentary exist. In the field of art seems to exist suitable concepts to solve this development problems. Particularly, the concepts to initiate classical dramas point out some interesting approaches we try to use. Currently, graphical issues will considered late while the website development. It results in inflexibility and cause problems for extension and change management. Our aim is to support the systematic and early involvement of layout and playout for that we develop an approach to *screenography*, which adopts and generalises dramaturgy and scenography. Scenography has its roots in theatre, film and television, i.e. outside the web area, and contains the composition of action space, plot and dramaturgy. The dramaturgy controls the sequence of scenes and determines the composition of information. Our claim is to show that WIS layout also requires scenography and dramaturgy to facilitate the understanding and memorisation of content and to support orientation within the WIS.

The following chapter introduces the fundamentals of layout development, particularly the development dimensions we have to consider for screenography. We try to clarify the aims and goals of these dimensions and demonstrate some different weightings we have noticed in practice. Further we propose a possible composition of dimensions for websites in future, that is based on a detected dimension hierarchy . In Chapter 3 we present a pattern-based approach of layout development, which helps to derive possible solutions from an intentional description. Therefore we analyse some types of patterns in detail. Within chapter 4 we introduce the grid development and propose a way of mapping the patterns to grids. Finally we conclude in section 6 and give an short outlook of further research directions.

2 Development fundamentals

This section gives an survey of layout development fundamentals. Layout development is affected by a huge set of parameters and conditions. Because it's possible to generalise this aspects, we distinguish six dimensions of layout development. In the following, we introduce these dimensions and discuss possible compositions. Moreover, we give some examples how different the weighting of each dimension can be, because the influence of the dimensions depend on application area. It clarifies, too, that it isn't necessary to consider all dimensions in each situation. From this, we derive a solution for websites we prefer in future.

2.1 Development dimensions

According to [ST05, Tha03] the development process is based on 6 dimensions, the intention, context, storyboard, content, functionality, and presentation depicted in figure 1. They are useful for high-level development, and we have to define their task and scope.

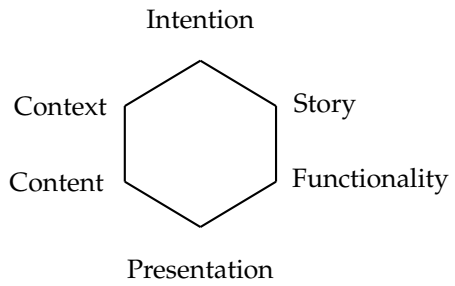

Figure 1: Development dimensions hexagon

2.1.1 Intention

The intention dimension specifies the type of an application we try to develop at a very general level. Therefore we analyse aims, goals, and visions and assign the application to a well-known and adequate application type. The classification is useful, because the difference of main objectives and their different weightings. For example, the major aim of information sites is the information and explanation while it could be a minor aim to sell some products. By contrast, the major aim of electronic commerce sites is to sell as much as possible products. A minor aim of such sites could be to give a detailed view about the offer to support the selling.

Different objectives implicitly influence the way of representation. Explicit, we are able to define the atmosphere resp. the ambience of an application. The ambience definition will only be done conceptional and helps to preselect colour-schema types. It doesn't specify concrete colours, because the intention doesn't consider cultural or religious preferences.
Another part of the intention dimension is the adaptation of layout and playout that is influenced by profile and portfolio of the users resp. the provider. The profile ascertainment is important for the preferences and behaviour patterns of the users. For instance we need it in the field of e-learning. As mentioned in [BZ04] there can be users prefer formal thinking and systematic working and others who are more example oriented. Thus, the profile of users influences the way of representation resp. the layout and playout in dependence on their preferences. The same is true for the portfolio [ST06b] of the users and the provider, which defines the tasks, involvement and collaboration. So, the type of a task or a bundle of tasks determines the way of representation. Moreover the playout is influenced by individual task collections, too, because they can be responsible for story changes.

2.1.2 Story

Because users doesn't act in the same way and their profile and portfolio is very heterogeneous, we need an individualised representation resp. several runs through the site. The story is able to provide only the needed content to avoid that the orientation of the users get lost in hyperspace. Mainly there are several specified stories within the story space. Thus, it is possible to handle different portfolios and demands, and so we are able to switch to other stories if they are more appropriate, in dependence on the availability resp. the rights and roles of a user. If we specify a concrete run through the story, we call it scenario. Typically, the task of a scenario is to guide the users.

2.1.3 Context

The environment is an important part of the context dimension. It isn't easy to grasp but often a problem while using a system. A part of context is the application context, e.g. the attention a user have or need to interact with the system. Further we have to consider the equipment, e.g. hardware and software resp. its abilities. Particularly, the usable bandwidth and the performance of a client strongly determines the way of representation. Moreover the rights a user have and roles are assigned to him influence the presentation resp. possible interactions. Besides, the quality of service (QoS) is a part of context dimension and can provoke that some parts have to be restricted to ensure the demands. For example a very fast response time could be ensured by removing details or support information.

2.1.4 Content

Content is the most important part of information systems, because it is the reason for build-up such systems. Of course, regarding the layout development the content dimension is very important too and concerns all presentable data as well as their types. However, often it is difficult to design an adequate database, handling the content in an appropriate way, because for example the structure of presented data typically differs from the organisation within the database.

2.1.5 Functionality

The functionality dimension is mainly represented by navigation and interaction aspects. Moreover integrity constraints and usable functions are a part of this dimension. Functions can have static or dynamic effects. We differ functions act local resp. cannot change the system from functions that are able to change the system or itself. Further we can distinguish internal and external functions. Internal functions realise, for example, the adaptation of the content to users preferences while external functions have to be perceivable by the users.

2.1.6 Presentation

The presentation dimension is influenced by the others and strongly determines the result of a development process as illustrated in figure 2 (development pentagon). The presentation dimension has the aim to concretise the definitions and restrictions of the other 5 dimensions. It completely defines the look and feel on the basis of these definitions. The representation is determined by the weight of each dimension and depends on the application area.

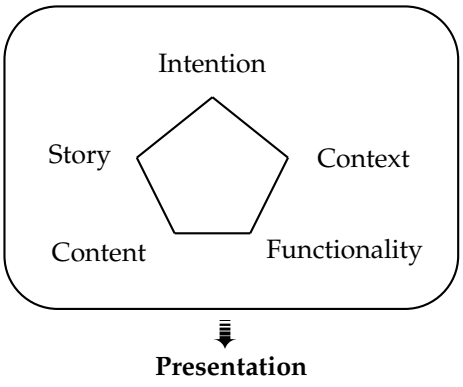

Figure 2: Development dimensions pentagon

2.2 Dimension composition

The specific composition of the dimensions plays an important role while the development process, because the influence of each dimension is determined by the application area. Therefore we analyse some typical processes and try to find an appropriate solution for website development.

2.2.1 Website development - state of the art

The classical website development considers the content, functionality, and presentation dimension. Mainly it starts with content and functionality specification and therefrom the presentation dimension will derived. In the case of predefined presentation templates it is necessary to integrate content and functionality, maybe realised by a downgrade of them. Choosing an adequate presentation template can avoid such downgrades but typically a template is developed for a specific application. Thus, if we start with the presentation dimension, it is essential to check content and functionality for adequacy regarding the chosen template, and if required, we have to inform the developer about integration problems. The remaining three development dimensions play a minor role while current website development. Often they will completely neglected. For instance, the intention isn't an explicit modeled part of a current development process. Often the involved persons hope that they all have the same imagination of the aims until the concrete realisation. Mainly well structured high-level commitments are missed but would be useful to find a real conformable main direction. Also the story and the context dimension will rarely considered. Ignoring the context can result in an inappropriate playout regarding the equipment of the target group. Further without a story specification we only clarify what should be a part of output but it disregards how to move within the website.

Figure 3: Website development dimensions - state of the art

2.2.2 Architectural development

Development in the field of architecture is hard affected by context, because typically buildings won't planned without consideration of it's neighbourhood. In general a good contextual embedding is a claim of inhabitants. Inside a building the context is important too. For instance a good orientation requires to satisfy the expectations of the main target group. We notice that architectural conditions can be demands of websites, too. Mainly the intention dimension has the top priority of architectural development. The intention represents the starting point and determines the main direction. Other dimensions, e.g. the context, are able to

overwhelm these definitions, but they shouldn't ignore intentional aspects. Only carefully selected parameters should differ from users expectations to avoid perception and cognition process impairments. However, sometimes it is helpful to disappoint expectations and arising the attention of the users. In the past, it has often been realised by animated-gifs. The third important dimension concerns the utility of the development. Therefore there is a functionality dimension that is able to influence the intention and context dimension.

As a result of intention, context and functionality determinations, other minor dimensions can influence the overall result. Mainly the content and story are parts of a second step. The story is important to ensure usability, e.g. museum tours should avoid crossing ways. The content dimension can have placement requirements and have to check the compatibility with decisions of the other dimensions. A third step of architectural development concerns the presentation. The presentation is a result of the definitions and restrictions of the other five. Nevertheless it is an important dimension responsible for the realisation in the form of an instance. Figure 4 highlights the main dimensions of architectural development.

Figure 4: Architectural development dimensions

2.2.3 Art development (classical drama & exibitions / museum)

A third way of development we can adapt from the art, particularly while looking to classical dramas. Main dimensions of art developments are content, story, and presentation. So it is the opposite approach of architectural development. The content of theatricals often is a given part. Also the story exists in the case of classical dramas. Hence, the presentation dimension is mainly determined by content and story definitions but not exclusive. Often there isn't a requirement to realise the presentation in a strict original way. So, the directors often try to combine classical themes with current events, wherefore the presentation differs from the original and reflects the directors view. In this case the presentation will influenced by secondary dimensions of art development, the intention, context, and functionality. So, if the director has the intention to combine classic and modern aspects, it will necessarily influence the presentation dimension. In the same way changes regarding the context or functionality can have effects on presentation.
This weighting of dimensions we detect regarding the arrangement within museums. Presentation, story, and content are the most important dimensions, while the other three dimensions are responsible for unique impressions resp. for individualisation of the supply. Online museums nearly have the same demands.

Intention	**Story**	**Content**	**Presentation**
	Context	Functionality	

Figure 5: Art development dimensions

2.2.4 Website development - state of science

While the development process, we noticed that there are bindings between the abstraction layer model (ALM) [Tha00] and the development dimensions, because of a hierarchy of the development dimensions and the possibility to assign the dimensions to the layers of the ALM as illustrated in figure 6.

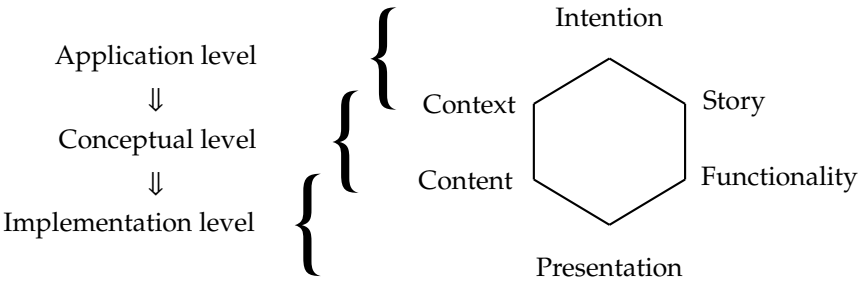

Figure 6: Dimension hierarchy

Because our main goal to generalise the layout development process we aim in start at the intentional level. Therefrom, we try to derive the story and contextual issues. Particularly, the context strongly determines the subsequent representation, because big differences regarding the equipment. The story specification helps us to get a good screen partitioning in dependence on story progress. With these high-level definitions we derive appropriate functionalities and specify the properties of content in consideration of the upper level definitions and restrictions. A last development step can be the presentation dimension that determines the layout in detail. At best the other dimensions lead to one possible result. Figure 7 illustrates the weighting we prefer for a general step by step website development allows e.g. flexibility as well as adaptivity.

Intention	**Story**	Content	Presentation
	Context	Functionality	

Figure 7: Website development dimensions (possible future scenario)

2.3 Development flexibilisation

[ST06a] introduces a way to develop web information systems and discusses the distinction of user and system perspective. It helps to understand the development process as a flexible, changeable progress. In dependence on application area and general decisions it's maybe useful to change the global specification order. Following this approach, it is possible to change the importance of development steps. Typically, early specified parts influences the following and so it can result in layout and playout changes, too.
Figure 8 depicts two different development approaches. The difference is the point in time when presentation issues will specified. Currently developers prefer the system perspective. By contrast, we specify the presentation in user perspective before the system concept so that the presentation development not depends to much on maturity of the information systems specification.

- A - Application domain
- Cp - Concept of presentation
- I - Implementation of presentation system
- R - Requirements prescriptions
- Cs - Conceptual specification
- IS - Information System

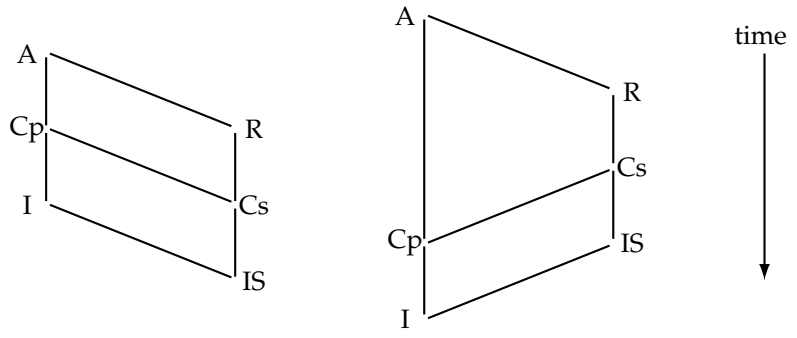

Figure 8: System ladder - user perspective (left), system perspective (right)

3 Pattern Development

We often have similar problems while development processes and prefer to find a universal solution for a whole problem field. Therefore, Christopher Alexander had introduced *design patterns* [Ale77] to solve some architectural problems by reusing concepts. Gamma [Gam96] picked up this approach to solve object-oriented software programming issues. Further, he said that patterns are useful for other application areas, too. So, we try to find patterns for layout development or in other words we are searching for presentation patterns.

"Each pattern describes a problem that occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice" - Christopher Alexander ([Ale77]).

3.1 Types of Pattern for Websites

In [MNST07], we distinguished between three principles taken from cognitive psychology, principle of visual communication, principle of visual perception, and principle of visual design. Visual communication deals with navigational and interaction aspects and aims in understanding as prerequisite of the communication partners. Visual perception resp. visual cognition concerns the orientation within the story and the screen. The objective of visual design resp. the composition is to realise a sufficient placement of media objects so that users have a good orientation within the arrangement and there are no problems to perceive the needed content.
We generalise these three principles as main parts of layout pattern development and enrich the classification in figure 9 by work progress and kind. The kind distinguishes between detected pattern types characterising a specific solution, e.g. there are many different types of evolution pattern underlie the same global restrictions. The work progress is important, because patterns are not only represented by static states. So, it can be helpful to define the behaviour of some steps within the story and to derive a composed pattern.

3.2 Pattern class: Composition pattern

The pattern approach was developed to allow the reuse of concepts regarding frequently occurring problems, wherefore each pattern describes a general solution. However, it is hard to decide for a pattern if some appropriate exist but there isn't a most appropriate one. In that cases, we maybe prefer to combine the most attractive parts of several patterns resp. create so-called composition patterns.
If we compose patterns, we have to consider which patterns dominate others,

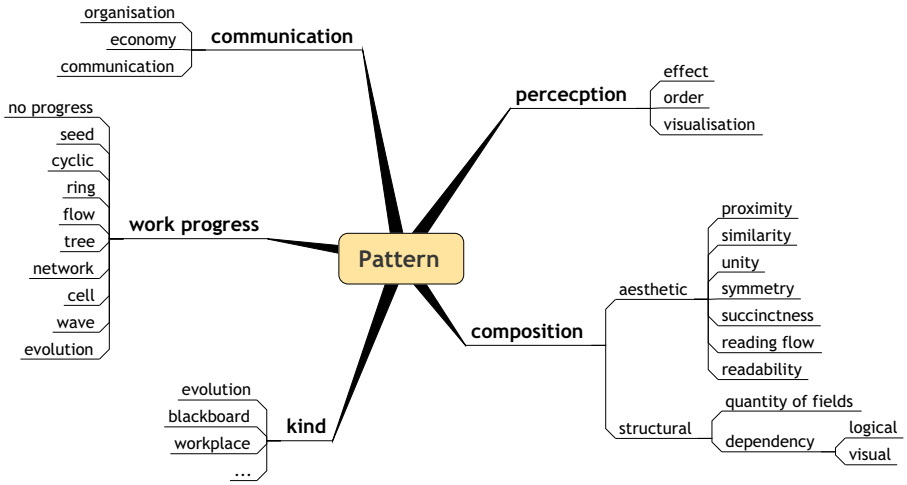

Figure 9: Pattern classification

because of partially existing dependencies between them, e.g. by introducing hierarchies.
Further, the presentation development distinguishes two kinds of composition patterns. On the one hand, we are able to compose patterns of different main categories (figure 9). These compositions are very general and their hierarchy determines the general decision order. Possibly, communication demands are more important than work progress demands, for instance, in the case of devices that are not able to realise some types of communication and therefore some progress types aren't available. On the other hand, we detect the composition of patterns, which are a part of the same pattern category. Such compositions have to consider not only the priority of the chosen patterns but also visual dependencies.

Within this preprint, we will introduce the latter kind of composition and use the screen partitioning as an example to get some output units. We declare output units as categorised, logical units of content that we try to present within the presentation space.

Typically, a presentation consists of some basic output units that we prefer to combine in a way the users expect, considering logical bindings as well as aesthetic aspects. Consequently, basic composition patterns contain a selected number of output units and define rules how to compose them. Therefore, it can be useful to analyse the output units of existing solutions with the aim to detect dependencies between them. At this development level we are interested in general dependencies and general placement problems, because specific content placement problems are a part of lower levels (grid level).

In general, it is possible to distinguish between *aesthetic* and *structural* composition. The main objective of aesthetic composition is the realisation of an adequate

representation by emphasizing structural aspects. We distinguish the following aspects:

- proximity
- similarity
- unity
- symmetry
- succinctness
- reading flow
- readability

The structural composition have to ensure the logical order and structural bindings as well as separations. We distinguish the following parts:

- quantity of fields
- dependency

The quantity is a very general part of content composition, and it compares the number of possible placement areas with the required output units of an application that shall be developed. So, there is an easy chance to check for adequacy, because the pattern specifies the maximum number of output units.

On the other hand, the structural composition is influenced by dependencies between the output units, that we try to assemble. Categories specify the type of output units and are required to detect these dependencies. In general, we distinguish visual and logical dependencies. Visual dependencies are determined by perception rules, e.g. sizes or the colouring of output units, while logical dependencies concern the order of output units to support the navigation process and create easy perceivable paths. Both aspects should be based on the story, because otherwise the coherence can get lost. If there are differences between the utilisation and the composition, a WIS could be hard to use.

A minimal playout only contains the *content* category, but usually there are two more main categories - *content* and *navigation*, supported by some style areas like the header. Often, additionally exist some categories like *support*, *help*, *details*, or *advertising*. Typically, there are restrictions regarding the coupling of output units that we have to consider while the development process. Therefore, a free placement of detected output units isn't recommended and we need rules how to compose them. Subsequently, we give examples that discuss the reasons for considering structural dependencies.

Case study example illustrating logical dependency:
Examplarily, we have three output units called *functionality*, *main content*, and *escort content*. We can ensure a general assignment ability if a derived presentation solution consists of at least three output areas. Thus, an instance of a two-column grid can't be an option in the following resp. at a lower development level. However, it is possible to decide for a common three-column grid as illustrated in figure 10. Within our example, we define that the functionality determines the content while the escort content gives further informations to the main content. So, the illustrated category assignment in figure 10 should be the expected representation.

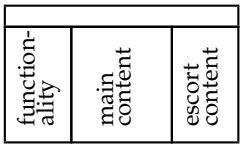

Figure 10: three-column grid - appropriate assignment

Other assignment patterns are imaginable, but we notice that not each possible solution is useful in practice, e.g. as illustrated in figure 11. The effort is to high to collect all adequate assignments per presentation pattern and it depends, for instance, from reading direction as a part of perception pattern as well as alignments of a specific solution. Nevertheless, it is possible to limit the choice if the dependencies between output units will collected. Subsequently, these dependencies can act as pattern rules and we are able to derive a limited set of appropriate solutions.

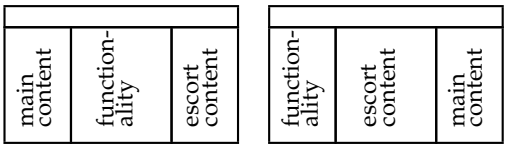

Figure 11: three-column grid - inappropriate assignment

Within the example, we detect two main dependencies between the output categories. On the one hand, there is a rule concerns the relation between content and functionality. Because the functionality determines the presented content, it is useful to perceive it before the content area. Further, they should presented close to each other, without intermediate parts, to perceive the coherence. On the other hand, the escort content is a subordered part of main content. So, we should perceive the main content before the escort content, too.
In general, we can ensure such demands by directing the placements. Later, we can refine it by fixing a reading direction. The reading direction itself is only an abstract part of composition patterns, because we are only interested in dependencies and general rules. If we apply these rules to the three-column grid consists of three lined up placement sections and check it against adequacy, we notice that

there are only two results that comply both rules. The first one corresponds with the appropriate solution (figure 10) while the second is it's inverse. The remaining assignments are only conform with one of the rules. Afterwards, we are able to recheck the result set and join several pattern categories. For instance, the cognition pattern clarifies the reading direction in detail wherefore one of the results becomes improper. For example, we can declare that we analyse information from top to down, from left to right, and from near to far. Therefore, only the first (figure 10) should be a possible solution.
If we analyse the binding detection process, we notice that it is urgent necessary to consider semantic aspects. In our case the escort content is assigned to the main content, but it is possible that it escorts the functionality, e.g. if we need a little bit more safety in using functions. In such cases the dexter solution of figure 11 is appropriate, while figure 10 is inappropriate. Moreover, if the content determines the functionalities, the sinistral solution of figure 11 is appropriate.

The dependencies described above only consider the logical order to ensure the usability. Often, it isn't sufficient to pay attention to logical dependencies, because the impression depends on the visual importance of an output unit. Therefore, size and contrast relations resp. visual hierarchies are important, too. Typically, visual dependencies support logical dependencies.

Case study example illustrating visual dependency:
Once again, we define three output units – *functionality*, *main content*, and *escort content*. We declare that the escort content have to support the functionality so that there is more safety in using functions and it's easier to get the right content. Moreover, we reuse the standard three-column grid that we mentioned above and ensure the correct logical order of the output units. Figure 12 shows that other variants of a three-column grid are possible.

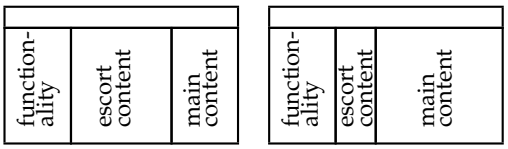

Figure 12: visual hierarchy - tile size differences

Originally, the standard three-column grid was developed for standard assignments like *functionality - main content - escort content (fig. 10) and* functionality - main content - advertisement. Accordingly, the sizes of the output units were developed for these assignments, too. For this reason, we perceive the sinistral solution of figure 12 as inappropriate regarding our chosen category definition. The main reason is the dominant visual importance of the escort content in comparison with the functionality area. Further, the main content has a subordinate visual role compared with the size of the escort content area.
The dexter solution of figure 12 is a possible realisation that considers the logical hierarchy of the output units and derives the visual importance. In this case, the size relations of the output units strictly follow the logical dependencies.

Not in all cases, it is useful to support the impression by size contrasts. Sometimes, it is better to build a visual hierarchy as depicted in figure 13. In such cases, it should be easy to perceive the information in the applicable order. Even, if there isn't a strict order as illustrated in the dexter figure.

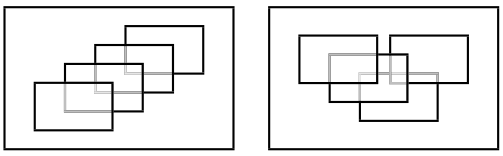

Figure 13: visual hierarchy - stacks

Moreover, it is possible to combine several approaches as illustrated in figure 14. The figures enrich the stack approach by different sizes to force the general stack impression using three-dimensional aspects. Each general stack causes a near far and a top down impression. The sizes of output units of the sinistral solution decrease top down, causing a near far impression, too. So, both approaches provoke the same hierarchy direction and an uniform impression.
By contrast, the output unit sizes of the dexter figure decrease bottom up. Consequently, the hierarchy direction of the size aspect differs from the stack impression. In general, it is useful to avoid contrary solutions, because they are a cause of perception problems. Therefore, the dexter solution isn't appropriate to realise a clear perceivable visual order.

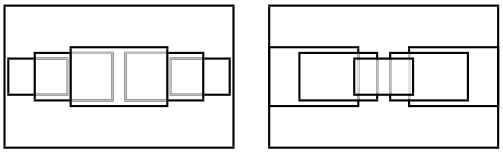

Figure 14: visual hierarchy - stacks & sizes

Altogether, we can find adequate representations if we consider logical and visual dependencies. At the best, the representations fulfil all demands of both dependency areas. Otherwise, we should search for the most suitable solution.

3.3 Pattern class: Progress pattern

Progress patterns describe possible types of movements that should be considered while the application development. The best-known progress was detected within the story dimension. [Tha03] had introduced scenes and dialogue steps to specify the story in detail. Further, scenarios were defined to characterise possible resp. appropriate ways through the story. Their topology describes a progress in detail. Nevertheless, progress patterns are not only a part of the story dimension, but

a part of all development dimensions that were introduced in chapter 2.1. The weighting of each dimension depends on the application and specified development targets, again. Figure 15 depicts the development dimensions regarding the work progress. Within this report, we are mainly interested in presentational aspects resp. progress patterns regarding the presentation dimension. However, we have to consider the other dimensions to benefit from their results.

Progress Intention	Evolution Story	Evolving Content	Work Progress Pattern
	Evolution Context	Evolution Functionalities	

Figure 15: Work progress dimensions

A simple step of searching for an adequate progress pattern only concerns a static composition within the presentation space. It's a good solution in the case of common arrangements, but in general, we need an extended view, that considers the specified story, the way of communication, and perception aspects, too.

Typically, semantic units of information are detectable as a part of the story dimension, e.g. a subset of dialogue steps as a part of a scene. It is useful to consider these semantic bindings for presentational aspects, too. Often, it isn't a desire to show all parts of a presentation unit at the same time, within the same screen, e.g. to realise a stepwise progress. As in the case of stories, it is very important to perceive all steps of a progress unit as related. We are able to emphasise such bindings by representing presentational elements of a progress unit in a similar way. Otherwise the coherence can get lost. Maybe, we further need a presentational separation if there is more than one progress within the presentation space. Because scenario topologies determine the type of progress, we have to analyse them and try to detect similarities. Further, we try to classify the types of progress considering known separations. Picked up the segmentation of [Bro00], we differ the following types of progress, extended by the *no progress* type:

- no progress
- network
- cell
- ring
- cyclic
- evolution
- flow
- seed
- tree
- wave

The simplest type of work progress is *no progress* (Figure 16*a*). It isn't a hypothetical type, because sometimes there isn't a need for interactions as well as changes regarding the presentation. Also there are possibly long-time changes so that a progress exists but the users cannot perceive it because of the temporal distance. For instance, in the case of electronic billboards, typically, there is a wish to separate the adverts so that it isn't possible to perceive a relation between these. Otherwise the advert-firm relation could be affected.

Sometimes there are no restrictions except that there have to be a progress. We call this progress type *network* (Figure 16*b*). It is the first real and most flexible progress type, which is hard to overlook for users and is appropriated for general parts of presentation to improve the accessibility by cross-links.

If we detect some decompositions in addition, we speak of the *cell* type (Figure 16*c*). We need it for refinement, because it can be necessary to differ the way of presentation at several levels.

Iterative additions to the network type result in the *ring* type (Figure 16*d*) and if it is directed, we call it *cyclic* (Figure 16*e*). If cyclic types are further affected by advancements, it is an *evolution* type (Figure 16*f*). For example, the cyclic type is useful in the case of offering online museum tours, because typically a round trip is closed and directed.

If a work progress with interactions is directed and moreover affected by advancements, we are able to describe a *flow* (Figure 16*g*) as another progress type. If there are duplications in addition, we call it the *seed* type (Figure 16*h*). This type of progress occurs in the case of collaborations. By contrast, if there are decompositions in addition to the properties of a flow, we call this type *tree* (Figure 16*i*), which could be helpful regarding retrieval. Besides, a flow with decompositions and duplications means the *wave* type (Figure 16*j*).

The figures depicted above in a schematic way distinguish some types of work progress and depends on factors of influence. So, the types are a result of the concurrence of these factors. Table 1 depicts the influence of the factors on the types of work progress.

	no progress	seed	cyclic	ring	flow	tree	network	cell	wave	evolution
directed	-	+	+	-	+	+	-	-	+	+
decomposed	-	-	-	-	-	+	-	+	+	-
iterative	-	-	+	+	-	-	-	-	-	+
evolutional	-	+	-	-	+	+	-	-	+	+
duplication	-	+	-	-	-	-	-	-	+	-
interaction	-	+	+	+	+	+	+	+	+	+

Table 1: Factors of influence

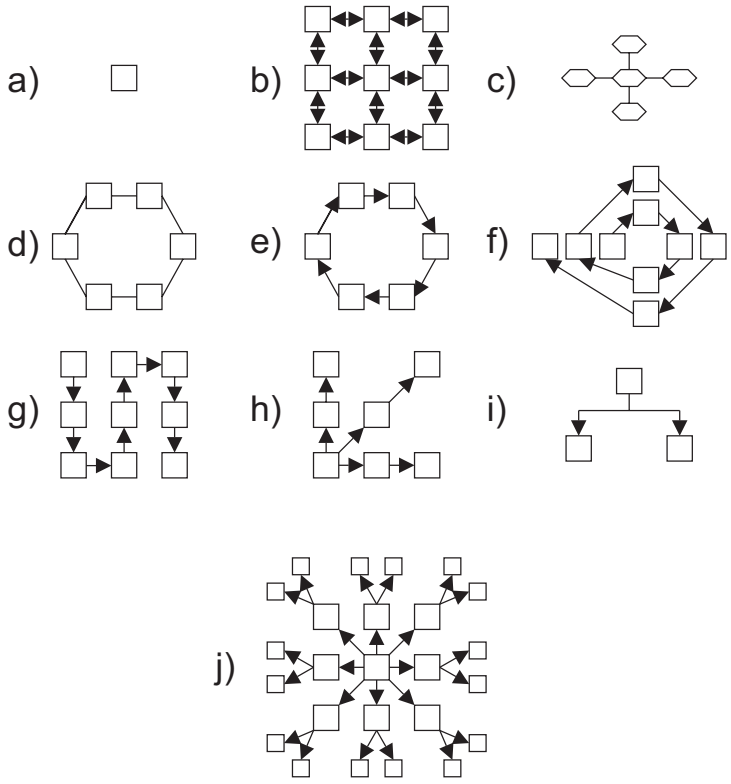

Figure 16: Work progress

Within table 1 we noticed some commonalities between the types of progress. Because they aren't obvious, figure 17 point these out. Therefore, we start with the very general *no progress* type and derive from this other progress types by considering further factors of influence. So, we get some progress types by a stepwise refinement resp. restriction.
Figure 17 only shows one possible transition graph to catch all described progress types. We notice that other transitions are possible, e.g. the grey coloured transition from the *seed* to the *wave* type. Further, if it is necessary, we are able to enhance the classification.

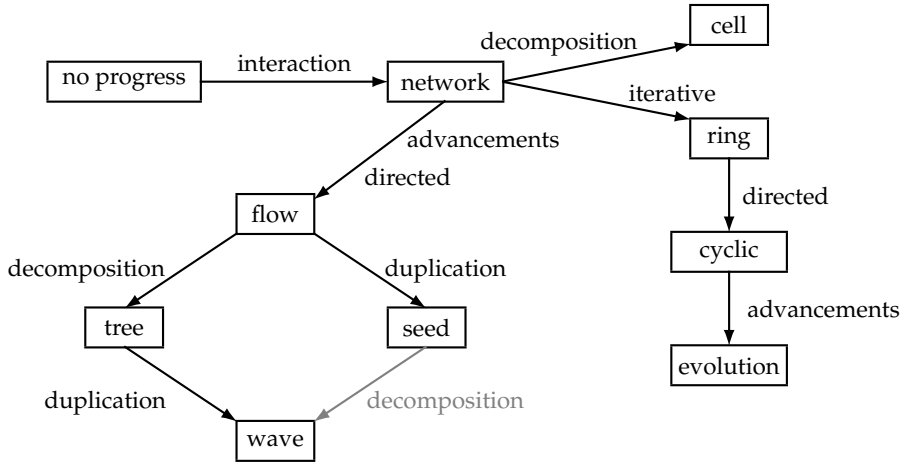

Figure 17: Work progress commonalities

3.4 Pattern class: Pattern kind

Because of different aims, goals, and types of content, there is a huge and unstructured set of demands against the presentation. While searching for an adequate presentation, we noticed that it could be profitable to classify the demands by the type of application. Therefore, it should be possible to detect similarities regarding the realisation as a result of the same demands in general. Further, we have to extract these similarities and have to generalise and classify them. Consequently, a *pattern kind* concentrates some typical properties of an application area and is responsible for the realisation of an easy reuse of existing patterns. So, it should be possible to speed up the assignment to possibly appropriated solution spaces. Towards the development of WIS, it means to consider the type of a website. At present, we distinguish between the following six main types of websites:

- information sites
- identity sites

- community sites
- education sites
- e-commerce sites
- government sites

All of them have individual demands in general but their derived representations must not be fully disjoint. So, it is useful to introduce further subdivisions, which are a little bit more specific and maybe applicable to more than one general type.

3.4.1 Evolution pattern

In [FFG96] *lifestreams* were introduced as time-ordered streams of documents, with the aim of organising and ease the finding. We try to generalise this approach with the help of patterns, in particular the evolution pattern. This pattern type tries to organise general output units, not only documents.
Evolution patterns are mainly represented by a perceivable progress. To achieve a progress we need parts of history and future or at least one of them, because it's very important to perceive more than one part at the same time. In opposition to the lifestream approach the evolution patterns must not be hard time-based, because for instance in the case of a task list the ordering is much more important than the time. Further it's urgent necessary that the ordering is perceivable, because otherwise problems occur as illustrated in Figure 18. The arrows within the illustrations point out that the users can have two different lines of vision, wherefore this example representation (grid) is inappropriate to induce an evolutional impression.

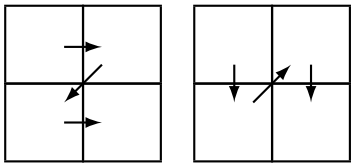

Figure 18: Lines of vision

Additionally, evolution considers visual bindings and a hierarchy between the output units exists. We have to consider them, because their importance for an perceivable visual progress. To support the perception it is helpful to avoid hard and abrupt direction changes regarding the line of vision. Typically, one output unit of such evolutional arrangements acts as present part. It can and should dominate the presentation to avoid perception problems. Only in the case of overview representations it's possible that we don't need it.

Evolution is more than a visual progress, because it considers the progress of the story, too. Typically, we mean with a progress the story progress, but we expect, that a well developed application supports it adequate by a visual progress.

Considering these, we are able to derive the following demands an evolution pattern should satisfy:

- Coordinated, sequential dialogues
- Mainly sequential navigation
- Specified reading direction
- Perception of history and future
- Similarity of neighbouring dialogue steps
- Proximity of neighbouring dialogue steps
- Symmetry of neighbouring dialogue steps

4 Grid Development

In spite of using pattern descriptions it is usually possible to derive more than one presentation result. We aim in finding the most suitable structure and it could be useful to collect different structures and describe and categorise them. Therefore in [MST05] grids were introduced.

4.1 Grid definition

Grids were adapted from (conventional) graphic design and used for organising page layouts, e.g. newspapers, magazines and other documents [vDLH02]. Mainly the organising is based on a tiling of the action space. A common way to do this is a disjunct tiling using grid points. In general, we divide the horizontal and vertical axes using grid points $x_{min} = x_0 < x_1 < ... < x_m = x_{max}$ and $y_{min} = y_0 < y_1 < ... < y_n = y_{max}$. A tile is defined by a rectangular region $\left[x_i, x_j\right] \times [y_k, y_\ell]$. Then we use a partition of the whole screen into tiles. A simple grid that only divides rows and columns, without any other restrictions is described in the following. It uses just 4 horizontal grid points $x_0 < x_1 < x_2 < x_3$, and only 3 vertical grid points $y_0 < y_1 < y_2$.

$$\begin{aligned} \text{up} &= [x_0, x_3] \times [y_1, y_2] & \text{left} &= [x_0, x_1] \times [y_0, y_1] \\ \text{middle} &= [x_1, x_2] \times [y_0, y_1] & \text{right} &= [x_2, x_3] \times [y_0, y_1] \end{aligned}$$

Usually, the “up” tile is used for some menu bar, the “left” tile for navigation links, the “middle” tile for major content and the “right” tile for side options. A visual result of this description is depicted in Figure 19.

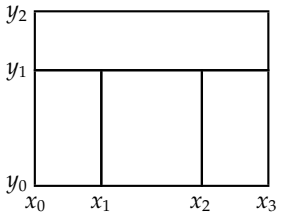

Figure 19: Common tiling

More complex examples we can describe in the same way, e.g. the fibonacci grid of [Mor07]. More sophisticatedly, the size of visual building blocks can follow a rhythmic structure that can be expressed by a sequence of positive integers. Then an observer perceives larger tiles of a sequence as being more important, in particular, if the sequence shows a monotonic pattern.

$$\begin{aligned}
\text{head} &= [x_0, x_9] \times [y_5, y_6] & \text{nav_spot1} &= [x_0, x_4] \times [y_4, y_5] \\
\text{nav_spot2} &= [x_4, x_5] \times [y_4, y_5] & \text{nav_spot3} &= [x_5, x_6] \times [y_4, y_5] \\
\text{nav_spot4} &= [x_6, x_7] \times [y_4, y_5] & \text{nav_spot5} &= [x_7, x_8] \times [y_4, y_5] \\
\text{nav_spot6} &= [x_8, x_9] \times [y_4, y_5] & \text{nav_active} &= [x_0, x_9] \times [y_3, y_4] \\
\text{square1} &= [x_1, x_2] \times [y_1, y_2] & \text{square2} &= [x_2, x_3] \times [y_1, y_2] \\
\text{square3} &= [x_1, x_3] \times [y_0, y_1] & \text{square4} &= [x_0, x_1] \times [y_0, y_2] \\
\text{square5} &= [x_0, x_3] \times [y_2, y_3] & \text{square6} &= [x_3, x_9] \times [y_0, y_3]
\end{aligned}$$

The result of this description is depicted in Figure 20. It shows the horizontal coherence of tiles between y_4 and y_5. In this area the tile size arises from the left to the right by $\frac{1+\sqrt{5}}{2}$ (fibonacci numbers / golden section). So it’s illustrating a progress. Moreover the square tile sizes rise up in the same way, but because of the composition it is a spiral impression of increase.

The very simple grid descriptions above don’t contain any information about the tile sizes resp. details of intervals between the grid points. They are necessary to describe complexe grids, because otherwise the expected impression of the grid could get lost, e.g. as a result of bad scaling. Moreover the defined behaviour of all grid tiles is important for the generation process. Maybe the equipment isn’t able to scale the grid, what should result in taking another grid. Further we are interested in bindings between grid tiles. We need them to derive appropriated grids with the help of some pattern descriptions. Both examples make clear that derived grids are only possible results. To complete the development process we need further detailed specifications on grid level.

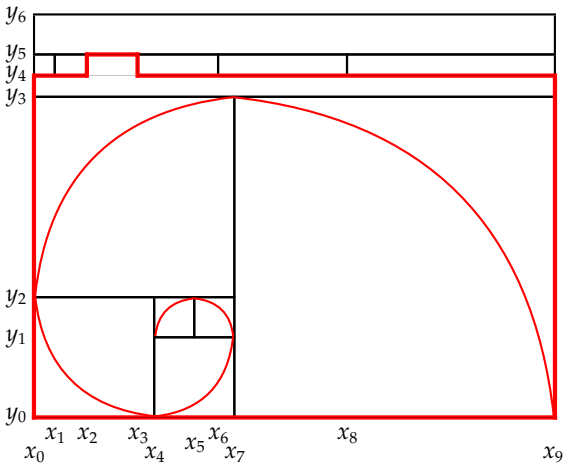

Figure 20: Fibonacci tiling

4.2 Grid generation

The previous chapter (4.1) has introduced grids as an approach for screen partitioning. In our opinion, well designed sites aren't a result of coincidence. So, we need methods for creation and adaptation of grids, because we are not interested in any grids, but in appropriate grids regarding the demands of an application. Therefore, we pick up some knowledge from the art, e.g. aesthetic and placement aspects of traditional design, and try to derive appropriate grids.

As mentioned above, the grid structure organises the screen into single segments that are connected in mathematic, geometric and aesthetic relation between each other and the screen.

There are different kinds of grid arrangements:

- linear grid with consistent grid structure
- fluid grid with flexible grid structure

While a linear grid shows a consistent structure with identical elements and constant navigation principles at all levels of interactive use, the fluid grid is based on a flexible structure with constant navigation principles with dynamic order, constant colour coding and repeated elements of a constant functional scheme.

Evolution patterns represent the development of interactive use as a multidimensional information flow and thus describe the userspace as a part of the special information and architecture of use. In order to achieve optimal orientation in the users process we believe it is necessary to adapt the development of visual design

grids to human experience and to use the conventions of space experience. Consequently, the human scale is the basis for the grid generation, comparable to the classic architecture. The human scale is related, in addition to the space dimension, to the sensory perception of spaces and is defined by the space categories

- impression space
- interest space
- action space

The valuation of the userspace includes not only the individual atmospheric dimension, but also the social and communicative component and refers to:

- quality of impression
- challenging character
- conciseness
- adaptation

This classic Human-Space-Relation contains the following aspects:

The formal aspect is related to the aesthetic of the userspace as an expression of a special shape and colour design The functional aspect is related to the specific task of the userspace and thus to its power and virtue.

The power of the userspace is valuating the actions performed, while the impression is based on the spacious cognition and reflects the emotional aspects of the space experience. Architectural structure is always aiming to harmonise the objective requirements to the using purpose with the emotional and atmospheric effects of a space. Consequently, there is a close relationship between the design of userspace and informationspaces of interactive information systems and the classical architecture. Not only the efficient use, i.e. organised action, short distances and timely optimised processes, but also the identification of the user with the space is an important premise for experiences and a positive feeling that is motivating the user to act.

4.3 Grid composition

Each developed grid has a task and is appropriate for selected application areas. If we analyse the structure of existing solutions, we detect some similarities. As in the case of patterns, it is hard to decide for a grid if some appropriate exist, but a most appropriate grid doesn't exist. Therefore, it could be helpful again, to combine the most attractive parts. Thus, we are able to reuse well-known structures resp. can compose selected of them.

Composition types
If we want to compose grids, we have to consider the type of their composition. Typically, we are able to distinguish the following grid composition types:

- Mixing
- Attachement
- Integration

The mixing type is very powerful and allows to develop new grids that are based on attributes of several initial grids. We need this type if the initial grids cannot combined directly by adherence or integration. In that cases, only selected attributes of the grids will combined, wherefore the appearance of the result can be completely different. However, if we want to derive such grids automatically or semi-automatically, we need a hierarchy, which defines which attributes of a grid dominates others.

An easy chance to compose grids offers the attachement type. In that cases, all initial grids keep their properties, because interactions don't exist between these. Grids of the attachment type cannot influence other grids, because as a reault of composition they are at most in touch with other grids or loosely coupled.

The integration is the third and most important type of composition. We need this type to embedd grids within a main grid. In such cases, the properties of the embedded grid have to follow the guidelines of the grid which embedds.

Composition areas
If we compose grids, we have to consider which patterns dominate others because of partially existing dependencies between them, e.g. by introducing hierarchies. Usually we have to consider global guidelines as corporate identity demands, wherefore it is useful to introduce a hierarchy. Because the corporate identity influences the whole representation we distinguish moreover between:

- grid frame
- grid body

The frame of a grid we understand as a global part that specifies, what should mainly presented over-all pages. Thus, such parts of a grid normally take a part in corporate identity determination. Grids of the body have to consider the definitions and restrictions of frame grids and are responsible for local decisions and local positioning, in other words for local representation issues.

Composition integration effects
Further, if there are grids, which integrate other grids, they should be able to control the behaviour of the subgrids. So, it is necessary to differ allowed operations. The composition behaviour of grids we can describe with:

- Open
- Restrictive
- Closed

Open means that the grid allows any type of subgrid. Maybe, a subgrid can influence the integrating grid, e.g. the size or the appearance. Restrictive grids allow subgrids with at least the same restrictions like itself. If a grid is specified as closed, then there are no subgrids allowed.

All grids shouldn't understood as applicable only for a unique page resp. content output at one specific state. Because the existence of complexe grids, influencing more than one page, we distinguish between common and complexe grids. Sometimes we need this to make the progress perceivable or to support the impression. Figure 21 illustrates possible instances of evolution grid.

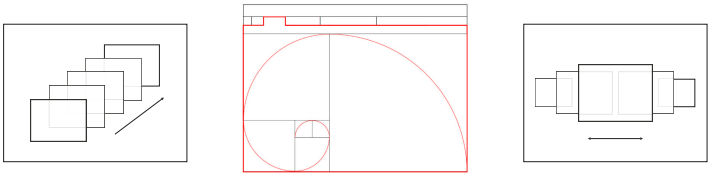

Figure 21: Evolution grids

5 Related Work

Website development has been the subject of several studies. As a result, some approaches and tools were developed. The presentation development is an important part of the website development and should typically result in an adequate layout and playout of content in consideration of user and provider aims.

[Sko03] discussed basic elements of presentation development, e.g. dots, lines, and shapes. Because each presentation is a composition of these elements, they are a prerequisite of presentation development. The perception of these elements is influenced by characteristics, e.g. size, colour, and contrast. So, the general impression of a user is influenced by both elements and characteristics, and it depends on user preferences, capabilities, and cultural as well as religious views. Regarding the impression of colours, Itten [Itt61] introduced an approach how to use colours. [Meg92] discussed impressions and effects regarding the style of letters, words, and images.

The composition of elements needs some rules, e.g. placement rules to order the content on the screen. The grid-based approach in [Gra98] tries to solve the placement problem by combinatorial analysis. The approach is possible because there are not too much sizes of basic elements and an overlapping isn't allowed.

At present, some general methodologies for WIS design exist, e.g. [HBFV03, Pat99, TD01]. All of them should be able to interact with aspects of presentation development. For example, within SiteLang we can specify some types of stories, which should be supported by an adequate layout as well as vice versa. Because we detected general and reusable parts as *work progress* and *composition*, we prefer an pattern-based approach, e.g. to interact with the specified story. Presentation patterns ease the development process by reusing concepts and are able to influence other parts of the WIS development process.

6 Conclusion

Layout development is a complex task because of the large variety of conditions and multiple choices to compose these. Often, the development of coherent pages is based on a general style resp. pages that are represented in the same fashion and provoke the same impression. Therefore, we need a systematic approach for the development of suites of web pages that can be easily used by a wider auditory. Software Engineering (SWE) has an understandable description of requirements to the SW that can be discussed with any stakeholder. SWE of presentation systems must also provide a way for description of the interfaces that should be developed. A pattern-based approach seems to be most suitable to accomplish the demands. Therefore, the paper has presented *screenography pattern* as an pattern-based approach of layout development.

Next step in future will be the specification of another patterns. Further, we will demonstrate in detail how to compose several patterns and how to benefit from the pattern interplay. Moreover, we have to specify the grids in detail developing an universal description.

References

[Ale77] Christopher Alexander. *A Pattern Language: Towns - Buildings - Construction*. Oxford University Press, 1977.

[Bro00] Laura Brown. *Integration Models: Templates for Business Transformation - The Authoritative Solution*. Sams Publishing, 2000.

[BZ04] Aleksander Binemann-Zdanowicz. Towards generative engineering of content-intensive applications. In *Principles of Software Engineering Conference (PRISE 2004)*, pages 41–49, 2004.

[FFG96] S. Fertig, E. Freeman, and D. Gelernter. Lifestreams: An Alternative to the Desktop Metaphor, 1996.

[Gam96] Erich Gamma. *Design patterns*. Addison-Wesley, 1996.

[Gra98] Winfried H. Graf. Constraint-based graphical layout of multimodal presentations. In *Readings in intelligent user interfaces*, pages 263–285. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[HBFV03] G.-J. Houben, P. Barna, F. Frasincar, and R. Vdovjak. HERA: Development of Semantic Web Information Systems. In *Third International Conference on Web Engineering – ICWE 2003*, volume 2722 of *LNCS*, pages 529–538. Springer-Verlag, 2003.

[Itt61] J. Itten. *Kunst der Farbe*. O. Maier, 1961.

[Meg92] Philip B. Meggs. *Type and Image: The Language of graphic design*. John Wiley & Sons, 1992.

[MNST07] Thomas Moritz, René Noack, Klaus-Dieter Schewe, and Bernhard Thalheim. Intention-Driven Screenography. In *ISTA*, pages 128–139, 2007.

[Mor07] T. Moritz. *Visuelle Gestaltungsraster interaktiver Informationssysteme als integrativer Bestandteil des immersiven Bildraumes*. PhD thesis, HFF Berlin-Babelsberg, 2007.

[MST05] T. Moritz, K.-D. Schewe, and B. Thalheim. Strategic Modelling of Web Information Systems. *International Journal on Web Information Systems*, 1(4):77–94, 2005.

[Pat99] Fabio Paterno. *Model-Based Design and Evaluation of Interactive Applications (Applied Computing)*. Springer, 1999.

[Sko03] David Skopec. *Digital Layout for the Internet and other Media*. Ava Publishing SA, 2003.

[ST05] Klaus-Dieter Schewe and Bernhard Thalheim. Conceptual modelling of web information systems. *Data & Knowledge Engineering*, 54(2):147–188, 2005.

[ST06a] K.-D. Schewe and B. Thalheim. Usage-Based Storyboarding for Web Information Systems. Technical Report 0613, Christian Albrechts University Kiel, Germany, 2006.

[ST06b] K.-D. Schewe and B. Thalheim. User Models: A Contribution to Pragmatics of Web Information Systems Design. In K. Aberer, Z. Peng, and E. Rundensteiner, editors, *Web Information Systems – Proceedings WISE 2006*, volume 4255 of *LNCS*, pages 512–523. Springer-Verlag, 2006.

[TD01] Bernhard Thalheim and Antje Düsterhöft. SiteLang: Conceptual Modeling of Internet Sites. In *ER*, pages 179–192, 2001.

[Tha00] Bernhard Thalheim. *Entity-Relationship Modeling - Foundations of Database Technology*. Springer, 2000.

[Tha03] B. Thalheim. Co-design of structuring, funtionality, distribution, and interactivity of large information systems. Technical Report 0315, Cottbus University of Technology, Computer Science Institute, 2003.

[vDLH02] D. K. van Duyne, J. A. Landay, and J. I. Hong. *The Design of Sites: patterns, principles and processes for crafting a customer-centered Web experience*. Addison Wesley, 2002.