The relation between design patterns and schema theory

Christian Kohls
Knowledge Media Research Center
Konrad-Adenauer-Str. 40
72072 Tuebingen - Germany
+49/7071/979-103
c.kohls@iwm-kmrc.de

Katharina Scheiter
University of Tuebingen
Konrad-Adenauer-Str. 40
72072 Tuebingen - Germany
k.scheiter@iwm-kmrc.de

ABSTRACT

Patterns capture the design knowledge of experts. But how is this expertise represented by the expert? When we mine for patterns, what is the ground in which we seek? Are there patterns in our head? And if so, how do the patterns in our head relate to the design patterns in the real world and the patterns we document? This paper tries to give some answers by referring to the principles of psychological schema theory. Schemas are some sort of patterns in our heads. A special type of schema, the problem schema, has many features in common with design patterns. The paper will discuss how schemata are organized in memory, how they are activated and constructed. At the end, we will discuss implications for the mining of patterns.

Categories and Subject Descriptors

D.2.10 [**Software Engineering**]: Design – *Methodologies*

General Terms

Design, Human Factors, Documentation, Theory

Keywords

Design Patterns, Schema Theory, Frames, Scripts, Mental Models

1. Motivation

Because design patterns are derived from real world designs it is assumed that the patterns are true as well [1]. The trouble is to ensure that a pattern has been captured in the right way. There is no automatic or formal procedure to mine a pattern. Patterns are fuzzy, and people may have different patterns in their head. To illustrate this, consider the concept of a "lecture" as a pattern. Most people understand what is meant by "lecture" and have their individual ideas what is meant by the term. The exact and intrapersonal essence of context, forces, problem, and solution is hard to capture, however. Some individuals may focus on a lecture's organizational issues, others may be more concerned about didactics. The point is that people have different patterns in their heads. As a result, the documented patterns would not only vary in style but also in content depending on the authors who wrote the pattern. Therefore it is necessary to distinguish between the patterns in the real world, the patterns in the mind of an individual and the documented patterns. This relation has been acknowledged in the pattern community, although its implications have rarely been discussed. Gabriel [2] distinguishes between real world and documented patterns: "A pattern, then, is both something in the world – the configuration found in excellent artefacts – and a literary form, that is, the written description of the physical configuration and why it should be build." The fact that patterns are part of the human mind is outlined by Vlissides [3]: "[…] people have had patterns in their heads for as long as there have been heads. What's new is that we've started naming the patterns and writing them down." This relation of pattern occurrences is illustrated in figure 1.

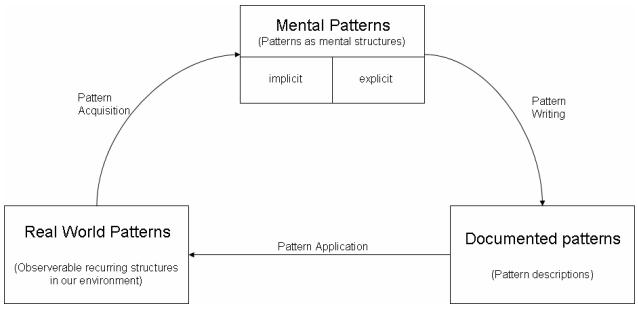

Figure 1. Real World, Mental, Documented Patterns.

Note that the concept of a "real world pattern" refers to the regularities in the structure of "real forms" in the world. It does not make any statement about the nature of reality, that is whether these "real forms" exist in an objective reality or in a socially/individually constructed reality. We only use the word "real" to explicitly distinguish between sensible things in the outer world and the things in our mind – the models in the world and our mental models. Pattern languages have already been considered as mental models that a designer has [4]. We are confident that schema theory offers further explanations of how patterns are represented in the mind and how they are acquired. Like patterns, a problem schema "allows problem solver to group problems into categories in which the problems in each category require similar solutions" [5]. A problem schema, too, captures the invariant parts of a problem and has slots that can be filled

with the specific properties of the problem (i.e. schema instantiation). As in patterns, there is a generic solution procedure attached to the problem representation, which is executed to produce a solution to the problem [6]. Linking schema theory to the pattern concept allows explaining many of the difficulties of finding the right level of abstraction, granularity, and detail in identifying the patterns.

2. Introduction to Schema Theory

"The captain asked the passengers to fasten the seat belts. They were ready to take off." Now, after reading the previous sentences did you have the image of an airplane at the airport in your mind? How is that as the text never mentioned an airplane or an airport? Obviously the propositions in the text allowed you to infer a situation that is more complex and richer in its details. This comprehensive situation is an interrelation of events and entities, and is stored in an internal data structure that can be activated by recognizing its typical features. Such data structures, or schemas, are mental representations in an individual's mind. Very different types of information can be stored in schemas, including physical objects, plans and strategies, behaviour patterns, or design knowledge. A schema bundles all the experiences within one class, that is, distinct experiences with similar features, and offers a generalized or abstracted representation. This process has been termed schema abstraction [7] or schema induction [8] in the respective psychological literature. It is characterized by extracting features that are shared by the experiences within a class and that are thus relevant to the schema as well as by abstracting away from irrelevant or superficial variations of these experiences. Thereby the resulting generalized representation allows one to integrate even slightly different new experiences within the same data structure.

In the following we will summarize the various aspects and flavors of schema theory while being aware that different research groups have quite different assumptions how schemas are achieved and function. Thus, schema theory is rather a theoretical framework that is specified in several ways.

2.1 Scientific origins of schemas

The term "schema" goes back to the old philosophers Plato and Aristotle [9]. In Plato's dialog The Meno, for both concrete and abstract concepts, a schema refers to the essential commonalities for example in music or shapes, or even in what makes up a brave man. Similarly, Aristotle speaks of form (or schema) when he means the essence or nature of the thing, that is, which basic properties and characteristics make an object distinct. In that sense the meaning of schema is already similar (if not equal) to our understanding of what a pattern is, since patterns always try to capture the core and invariants of concepts. For example, POSA-5 refers to the patterns an experienced expert can draw on as 'solution schemes': "Expert architects and developers can draw on a large body of these 'solution schemes' to address design problems that arise when developing new systems and maintaining existing ones" [10].

In this discussion we pay attention to how these schemas or patterns are represented in our mind. For Kant, a schema was the link from empirical information to the pure categories or concepts that he believed were given a-priori in the mind. Every perceived stimulus had a relation to an ideal concept, for instance the perceived chair can only be interpreted as a chair if it is linked to the idea of a chair. This linkage is done through schemas, which are representations of the perceived phenomenon. While psychologists sustain the idea that perceived information is linked to pre-existing knowledge in the mind, they no longer believe that this knowledge is given a-priori. Rather their interest focuses on the processes of how schemas are acquired, applied, stored and manipulated in memory. The term was established in psychology by Bartlett [11], much research has been done by Piaget [12,13,14] and Rumelhart and Norman [15]. Schema theory has been introduced to education by Gick and Holyoak [7], VanLehn [6], and Sweller, van Merriënboer, and Paas [16]. In the educational context, problem schemas are seen as the pivotal mental structure that is constructed by learning from examples and that guides problem solving in scholastic domains like mathematics or physics.

2.2 Schemas as structural units

A schema is a structural unit that represents a concept, situation, event, plan, behaviour etc. in a generalized form, that is, it contains an abstract representation of multiple instances of the same kind. In schema theory this unit is an internal data structure in the memory that organizes an individual's similar experiences. It is used to recognize similar and discriminate dissimilar new experiences, access the essential elements of the commonality (both verbal and nonverbal components), draw inferences, create goals, develop plans and utilize skills procedures, or rules accordingly [9]. For example, the schema of a CAR is a generalization of all the cars a particular individual has seen or experienced before. Though it does not contain all the details of any car seen, it contains the essentials, the core features and properties shared by (almost) all cars. If an individual sees an object that shares the same elements and relations as stored in his or her CAR schema, the individual will recognize it as a car. As such, the car appears as a gestalt, an encapsulated unit that appears as a whole form, in which the configuration of components is in full harmony. Gestalt theory is ubiquitous in schema theory and pattern theory. Both Bartlett [17] and Piaget [18] have been inspired by gestalt theory. Alexander explains our perception of whole forms with reference to gestalt psychologists [19] as well as when he talks about wholeness [20]. In POSA-5 [21], the "quality without a name" [22] is a synonym for wholeness in the pattern concept summary. This quality arises if a design is coherent as a whole. It is based on the view of gestalt psychologists that the sum is more than its parts (thus, it creates emergent features).

To get a better understanding of what holds a schema together as a unit, in the following, three structural features of schemas will be outlined. First, schemas are constructed based on variables and their associated values, whereby second, choosing precise variable values will lead to a concrete instantiation of a schema. Third, schemas can be used to represent declarative as well as procedural knowledge.

2.2.1 Variables and variable values

In schema theory, the building blocks of schemas are thought of as variables (sometimes called slots or attributes [23]). Each variable can take a (constrained) range of values (or slot fillers or

attribute values) as input. The possible space of variable values reflects the range of experiences a person has had with a specific concept and determines what new experiences will be immediately accepted as belonging to this concept, that is, without further need of modifying the existing schema. In some cases, especially if a schema is highly specific, a constant may be used as a filler rather than a range of variable values. For instance, for many people a car always has four wheels, whereas for others, who have had a broader range of experiences, the attribute 'number of wheels' may take values from 3 to many. As will be outlined in section 5, learning in schema theory often refers to modifying the variable values by either extending or constraining their possible range.

The variables that make up a schema are often interrelated, which is also why the configuration of their variable values is not independent. Imagine a DRIVING schema, where the CAR is a variable value as one can drive other vehicles than cars. The DRIVING schema may have other variables, such as driving style, street type, or target. Each of the variables may contain different values, for example the variable slot street may contain highway, country road, or jungle road. If one drives on a jungle road the options for driving styles are reduced as the ground does not allow speed rallies. Hence, variables are constrained by each other, and a specific configuration of some variable values implies that certain values are more likely for other variables. These constraints and probabilities for the occurrence of specific variable values are an implicit part of the schema structure. They have two important functions [15]:

1. The range of valid objects for a variable slot is given.

2. If specific information about the variable is missing, it is possible to make guesses taking the probabilities into account.

For example, there are different types of vehicles that can be driven; that is, a car, a bus or a balloon are all valid values for the DRIVING schema. However, one cannot drive a traffic light, an invalid value for the slot. Also, the valid values are of different likelihood. In the sentence "Bob was driving yesterday" the concrete vehicle is not mentioned. However, it is more likely to think that Bob was driving a car than to imagine Bob in a balloon, although this is a valid option. In the same way, the introductory example activated the schema of an AIRPLANE AT AN AIRPORT because it is most likely that the information given refers to that situation. There are other situations that could be meant, for example a speedboat making a trip with tourists or a child in a role play. Which values are probable is hence determined by individual expectations based on one's own experiences, or heuristics.

Variables and constraints are fuzzy rather than exact. Not every constraint has to be satisfied in order to accept a concrete instance as being represented by a schema, but not too many constraints may be violated either. For example, a car usually has four tires. But if a tire is missing, it still remains a car. However, if the form is too different from former experiences of what a car is, we will no longer perceive a car.

To summarize, schemas are organizational structures of memory that interrelate variables that frequently reoccur. A design pattern, too, is a logical structure that consists of variables [24]. The variables are highly interrelated and changing one design variable to make it fit to the requirements of the design problem can cause other variables to misfit (e.g., using high quality material to make a machine robust may increase the production costs too much). While the interrelations of variables within a pattern are strong, the interrelation of variables between distinct patterns is loose [25]. Hence, the configuration of variables within a pattern does not affect other patterns and thereby reduces the complexity of the design problem.

2.2.2 Schema instantiation

Filling the variable slots with exact values is called schema instantiation and its result is a specific mental representation of the abstract concept stored in the schema [26]. For example, while the CAR schema consists of the abstract representation of all cars, its instantiation refers to a specific car. Therefore, schemas also have a generic component that allows the individual to derive specific structures from the abstract structures. Not only can the individual retrieve the images of cars one has seen in the past by instantiating the slots with the right information, one can also imagine new cars. Or, to use a plan as an example, once the child has learned how to drink from a cup (it has acquired that schema), it can transfer that knowledge to any other cup or cup-like beverage containers. A schema understood in this way is not just a collection of experienced objects but an object space that allows you to recognize similar objects and mentally (re-) generate objects. Likewise design patterns are not only generic but generative as well. In the ways schemas form configuration spaces, patterns define design spaces that include all the designs that implement the pattern: "A pattern describes a coherent yet infinite design space, not a finite set of implementations in that space." [27]

The generation space of schemas is defined by the variable slots and their associated value constraints, including variables that are mandatory (that is they nearly always occur) or optional. In a specific configuration of the variables, objects that are experienced (i.e. recognized) or have been experienced in the past (i.e. retrieved from memory) are represented in the schema instantiation. Since the variables slots have different probabilities for their values, the object space implicitly includes a prototypical representation of the object. This prototypical representation is the schema instantiation in which all the slots are filled with default values.

In the way schemas constrain the valid values, patterns capture the invariants of good designs - that is the range of variable configurations that form a solution to a problem in a context. "We are interested in those links between variables which hold for all forms we can conceive. Not only statistical correlation but also causal relations" [28]. Because for any form the list of characteristics is infinite, one has to extract the variables that matter. The ones that are typical for the form, the ones that are typing the form. "The pattern is an attempt to discover some invariant features, which distinguishes good places from bad places with respect to some particular system of forces." [29]

2.2.3 Declarative and procedural knowledge

The data structures can capture declarative knowledge as well as procedural knowledge. Some authors distinguish between scheme (procedural knowledge) and schema (declarative knowledge), and

some translation errors have blurred this original differentiation by Piaget. However, the selection, instantiation, modification, creation and adoption of schemas apply equally for scheme and schema. Thus, in the ongoing text we will refer to both types of knowledge by the term schema.

Because schemas can represent both, objects or operations, a schema that consists of sub schemas can include both declarative and procedural knowledge. Both kinds of knowledge can be either directly included in a schema or referred to by variables. Because each schema has always optional external relations, it will point to various types of associated knowledge. Of special interest is the structure of problem and solution which will be discussed later.

By analogy, patterns capture declarative knowledge and try to externalize procedural knowledge in their solution sections: "The solution part of a good pattern describes both a process and a thing: the 'thing' is created by the 'process'." [22]. Furthermore, a pattern tells about a form not only what it is but also what it does [24].

2.3 Interrelations among schemas

Every schema is embedded in a hierarchical network that reflects the interrelations among schemas. These interrelations can take two forms, namely, (1) part-of relations that reflect the fact that each schema always is an ensemble of interrelated parts that co-occur in a network, where each of the parts is a schema itself [15], and (2) is-a-relations that introduce the idea of inheritance of structures into schema theory. Both types of interrelations are outlined in the following paragraphs.

2.3.1 Has-a/Part-of relations

The variables are the means of interrelated entities in schemas. A set of networked variables builds up a schema, but each of the variables represents a schema itself. For example, the CAR schema has a set of strongly coupled sub schemas such as WHEEL, TIRES, or MOTOR. A schema can be completely decomposed into its sub schemas, and each of the sub schemas has a part-of relation to the superior schema. Vice versa each schema is always a sub schema of a larger context. The TIRE schema for example can be part-of the CAR, but it can also be part-of an AIRPLANE. Similar, a CAR can be part-of a HOLIDAY TRIP or a DRIVE TO WORK. Thus, not only the internal elements of a schema are variable but also its external interrelations to superior schemas. In general, each schema A consists of multiple sub schemas B1…Bn. The schema A, however, is more than the sum of its sub schemas because it itself defines additional structure concerning the way in which B1...Bn interrelate. Also, each schema is part of a larger whole C. As a general rule we can note that schemas can be embedded into other schemas recursively.

Due to the aforementioned part-of relations activating a schema in memory will give access to two types of information by means of association. On the one hand, because the variable values of the current schema are schemas themselves, further information on these variable values can be retrieved from memory. On the other hand, because a schema may itself be part of other schemas, information on the latter can be accessed in memory.

Schemas can also be loosely coupled. For example, the schema MAINTENANCE is associated with a car, but if one sees a car one does not always think of maintenance. If, however, the car is rusty or makes suspicious noise, the association with maintenance is more likely; that is the variables condition and sound influence the activation of the schema MAINTAIN CARS which consists at least of the schemas CAR and MAINTENANCE. Hence, the association between MAINTENANCE and CAR is operationalized by the superior schema MAINTAIN CARS. Strong coupling, as found by the subparts of a car, is also given for CAR and DRIVING because both schemas are in most of the cases slot fillers for the DRIVE CAR schemas.

This hierarchical organisation of schemas is considered by some schema theorists as the complete cognitive structure of human behaviour [30]. It is a means-end structure in which an initial state is transformed into an end state by splitting up each problem into sub problems. At each level of the hierarchy there are alternative schemas possible to solve the problem. "Another scheme might be a tentative plan for the solution of a problem, which is characterized by the start conditions, an outline of the goal to be reached, and some ideas of the route of subgoals by means of which one will try to reach the goal" [31].

2.3.2 Is-a relations

This type of relations between schemas is given by different levels of abstraction. One can have a schema of a DOG while having a more specific schema of a POODLE, or a more general schema of an ANIMAL. If a person perceives a poodle, it can be represented by any of the three schemas appropriately. However, the schemas vary in detail and generality. While the POODLE schema only applies to a small subset of animals, it provides more specific knowledge about poodles, for instance what color the fur can have. The ANIMAL schema on the other hand applies to all animals but comprises only information that is valid for all animals. It is possible to switch the level of detail by which we consider an object. We can see the poodle as a poodle, a dog, or an animal. Schemas that are at the same level of abstraction, however, are competing: a dog is either a poodle or a dachshund, but never both. For any actual situation, schemas of different levels of abstraction are valid, but for any level of abstraction there will be one schema that is more appropriate than others. Similar to inheritance in object oriented programming, the more specific schemas inherit the properties and procedures of the more general schemas. Although not explicitly stated in the literature on schema theory, we can assume that multiple inheritance takes place since concepts can overlap. For example, a family dog may inherit the properties of dogs but also the properties of the schema FAMILY. Furthermore, we can say that the type is an important constraint for the variables of a schema. If a specific schema type is accepted in a variable slot, then the specialized schemas are valid as well. For example, in the DRIVE SCHEMA, there is the slot STREET. STREET itself is a schema, and we can fill the slot with any type of street, such as HIGHWAY, COUNTRY ROAD, or JUNGLE ROAD. These streets differ in their specific properties, for example surface materials, size or number of lanes, but on a more abstract level they are all streets and share the common properties of a street, for example the shape.

Alexander [32] observes: "Nature is always full of almost similar units (waves, raindrops, blades of grass) – but though the units of

one kind are all alike in their broad structure, no two are ever alike in detail. 1. The same broad features keep recurring over and over again. 2. In their detailed appearances these broad features are never twice the same." Like schemas, patterns are always generalizations leaving out the details. They abstract from concrete forms; and concrete designs are instantiations of the pattern. The level of abstraction is given by the number of features and relations that are taken into account by a schema or pattern. Like schemas, patterns can exist at all scales [33].

2.3.3 Hierarchic organization

Furthermore, the hierarchic organization of patterns is reflected in pattern languages [34] and logically explained by set theory [35]. The relationships between patterns have been classified in various ways. Noble [36] distinguishes between the primary relationships that "a pattern uses another pattern, a pattern refines another pattern, or a pattern conflicts with another pattern" and the secondary relationships (i.e. used by, refined by, variants, variant uses, similarity, combines, requires, tiling, sequence of elaboration) which can be expressed in terms of the primary relationships. Using a secondary relation adds more meaning to the relation by making it less general, for example the relations "next step" and "owner of" have certainly different meanings but at the same time are both of the type "has-a".

It is notable that in Alexander's pattern language [34] the only expressed relations are the uses/used by relations at the beginning and end of each pattern. This type of relation delegates problem-solution details to sub patterns and it shows that Alexander's emphasis is on the decomposition of complex problems. In fact, some of his patterns are related in different ways. HOUSE FOR SMALL FAMILY, HOUSE FOR COUPLE and HOUSE FOR ONE PERSON describe different house types. On the one hand, these patterns can be considered conflicting patterns [36]. On the other hand, one could argue that they are different refinements of an abstract pattern - because the houses are specialized solutions addressing particular contexts.

Comparing the schema and pattern concept, the Has-a/Part-of relations of schemas are uses relations in patterns, and the Is-A relations in schemas are refines relations in patterns. So, what about the conflicts relation? In a way, one could argue that patterns never are in real conflict because different contexts readjust forces for the same problem type and therefore require specific solutions, e.g. the different house patterns all apply to different contexts (families, couples or singles). Likewise the problem of transportation calls for different solutions (e.g. car, bus, train, plane) because of varying contexts and hence different weights on the forces [37]. If the problem statement is to create a website, the solution may look quite different depending on whether one creates a news site, a university portal, or an online community [38].

In addressing the same (core) problem, some pattern solutions can be in competition [21]. Also, patterns can satisfy forces on different scales [39]. In that sense patterns are in conflict, and they compete for implementation. In schema theory the conflicts relation is not an explicit relation type between schemas. Nevertheless, it is assumed that schemas compete for activation depending on their appropriateness as the next section shows.

3. Activation of Schemas

To understand a given situation and to plan an appropriate activity means to select an appropriate configuration of schemas taken from the currently available repertoire of an individual. The selection of appropriate configuration of schemas to account for the situation is called comprehension by Rumelhart and Norman [15]. This configuration consists of schemas that can be instantiated in a way that the current empirical data fits into slots of the activated schemas. The process of activating the right schemas is a complex pattern matching task in which memorized schemas cooperate and compete. Schema activation is triggered by the existence of relevant data. For example, the data "eyes" and "nose" can activate the schema HUMAN FACE, because it has the appropriate slots in which the given data can be properly assimilated. However, on a more abstract level there are competing schemas, such as ANIMAL FACE. Also, to activate a more specific schema MALE or FEMALE FACE, further information is needed. A name, "Sarah", could clarify that the schema FEMALE FACE should be activated. The three activated schemas EYE, NOSE and FEMALE NAME hence cooperate to activate the higher level schema FEMALE FACE. Schemas are considered to be self evaluating. That is, for any given input stimuli a schema can evaluate the likelihood that it can represent the empirical situation. The process of calculating such a fitness value, is executed by the schema itself – the algorithmic knowledge to generate output values according to input values is implicit given in the structure of the schema. To recognize the situation, the schema that best matches the features of that situation is selected. This is a fuzzy logic operation because the variable slots of the schemas are assigned with probabilities. So, if in the current stimuli some features are missing, unusual or additional, a schema can still be appropriate to represent the situation if in total the features match is better than in any other schema. Some feature settings, however, are not tolerated. For example, if the information "diameter of 5 cm" comes in, the schema HUMAN FACE has to be discarded.

3.1 Bottom-up and top-down

The process of schema activation works both bottom-up and top-down. Perceived information activates low level schema candidates at the bottom of the hierarchy, which produce output data on a higher level. For example, visual patterns activate specific shape schemas in this bottom-up process. This compound information is the input for higher level schema activation. If a schema is found in which the configuration of basic spatial objects is similar to those of an eye, this schema is activated and the output data is "EYE". This output data, again, is available in the recognition process. In combination with further data, such as "NOSE" and the word "Sarah", the FEMALE FACE schema is activated. From this schema, additional knowledge is inferred in a top-down process. For example, the schema knows that lips, ears, eyebrows etc. are most likely to co-occur. This knowledge can then be used to further interpret the available data on lower levels. In a comic drawing we can interpret a single line as a mouth if we have the context information that we see a face. This inference also helps to substitute missing information. If a text does not mention a mouth explicitly, we derive a default assignment from the MOUTH schema and fill the slot of the FACE schema. Hence,

we are likely to think of red lips, although green or blue lips are possible (if a lipstick is used).

The context of concurrently activated schemas and the current empirical perception are both input data. The later is encoded into schemas on the lowest level. A specific combination of lower level schemas makes it more likely for a specific schema on a higher level to be activated. This is due to the interrelations of variables. If the current lower level schemas are "better" slot fillers for schema A than schema B, then schema A will be more strongly activated than B. As a result, the activated schema A raises expectations about further variables. That is, on a lower level additional schemas become more likely to be found because they are expected from A. For any schema X, the activation depends on the available data. Positive stimulation can come from lower levels (X has parts that are activated) and higher levels of the hierarchy (X is part-of an activated schema). The strength of activation influences other schemas in their own evaluation. If, for some reason, the schema HUMAN FACE is denied (too many mandatory parts may be missing), this has consequences for other schemas that are already activated. The denial of the FACE schemas is negative input for the EYE and NOSE schemas, hence, they have to be re-evaluated. Maybe the shapes that first had been perceived as eye and nose have a different meaning.

3.2 Activation triggers output

The evaluation of a schema yields a certain level of activation, which in turn will affect the recognition of an observed object or scene. Strong activation means the individual has recognized a familiar situation. With its activation, all the interrelated variables get weight. Some of these variables can be operations, actions, and plans. Hence, the recognition of a certain situation also activates elaboration, planning, and execution knowledge. Again, this knowledge can be split up into sub schemas. To achieve a required state (as defined by planning and execution knowledge), the schema refers to lower-level schemas that have stored the knowledge how to achieve that state. On the lowest level, the schemas cause actions of the individual. Throughout this process, the schemas are constantly re-evaluated and if at one point there is a misfit, strategies have to be changed, that is errors have to be corrected. The schema itself and the cooperation of schemas (which are defined by the interrelation) can be improved by continuous learning (tuning).

4. Problem Schemas

In problem solving, two situations can be distinguished, namely, solving problems for the first time and handling recurrent problems [6]. If a specific type of problem is encountered for the first time, the problem solver does not have any knowledge (i.e. problem schemas) available to solve this problem. Thus, one can implement only problem solving strategies that do not require any prior knowledge, that is, so called knowledge-lean problem solving strategies. While these strategies may certainly fail in particular if dealing with rather complex tasks, they are the only ones available to the problem solver. On the other hand, if a person has already acquired knowledge on solving problems of a particular type, because the individual has made recurrent experiences with this problem type – either by learning by doing or studying illustrating examples – then knowledge-rich strategies become available. These strategies rely on applying existing problem schemas to the task at hand. Both problem-solving strategies will be outlined in the following paragraphs.

4.1 Solving design problems for the first time

Problem solving based on applying knowledge-lean strategies can be analyzed by considering two cooperating sub processes, understanding and search. The understanding process produces mental information structures that represent the problem according to the understanding of an individual. Its major outputs are two states, the current situation in which the problem arises as the initial state, and the desired situation in which the problem is solved as the final state. Solving the problem is a search process in which the solution is calculated or found by taking moves in a problem space. The problem space consists of (1) the initial problem state, (2) operators that can change a problem state into a new state, and hence allow moving in the problem space, and (3) the goal state. Each state in the problem space can be reached by a sequence of operator applications [6]. The difficulty of the problem is correlated with the topology of the problem space[40], though people may not be aware of how they step through the problem space. States in the problem space could be formally described by a state-representation language. For design problems, this language could describe the modeled form of the design and how it satisfies the given forces. If a state is found in which the form balances all forces, this state represents one solution. While looking for the solutions, the problem solver may get a better understanding of the problem itself. Hence, some of the insights gained are often changes of the problem space itself [41, 42]. For design problems this could mean that during the process of finding the right form, the designer finds additional forces that have to be taken into account. A state in the problem solving space corresponds to a set of assertions [6]. In the case of the designer an assertion means that a particular configuration of a variable has influence on the forces. Moving in the problem space is to vary settings of design variables to see whether the whole design better fits to the problem in the context. The operations of the problem-solving process then are the incremental changes to the assertions, that is the piecemeal variations of form. The challenge is that each variation influences more than one force and that some changes may lead to undesirable results. Hence, the operations can lead to dead ends. In that case, a backup-strategy can lead the designer back to a state which is closer to the solution. One can then proceed with another set of operations (that is change design variables). Heuristics, or rule of thumbs, are often used by problem solvers to decide which operations to apply in a state that satisfies certain conditions related to the heuristic. Finding a good solution is not random search. Rather, it is achieved by small transformation of states in a problem-space. The transformations are chosen based on experience of the individual. If the steps taken mean progress towards the solution, then the path is continued and the applied heuristic is strengthened. Finally, a solution state is reached.

Complex (design) problems can be decomposed into sub problems of smaller granularity. The hierarchical decomposition of artificial structures and problems is described by Simon [43]. It is notable that Alexander has referred to some of the earlier works of Simon [44] who has done much research on problem solving and is often referred to in literature on problem schemas. That the patterns in (software) design are likewise an approach to

decompose complex systems into a small number of recurrent subsystems as described in Simon's *The Science of the Artificial* [45] is highlighted by Grady Booch [46].

4.2 Recurrent problems and solutions

If subjects recognize the stimulus of a familiar problem, however, they do not seem to start the search process through the problem space. Rather, they retrieve a ready-to-use solution procedure and follow it. The knowledge unit that is available to solve problems of similar classes is called a problem schema [6]. Problem schemas are more likely to be applied by experts of a domain and allow for the implementation of knowledge-rich problem solving strategies. They consist of both, knowledge about the problem class (i.e. declarative knowledge) and the skills to solve problems belonging to that class (i.e. procedural knowledge). They are considered a specific form of schemas, which have the same characteristics as have been discussed in the previous sections about schemas. Thus, both parts of a problem schema are composed of variables that can take different values, which are used to represent a problem class' relevant features as well as operations to solve respective problems. Moreover, as suggested by schema theory problem schemas are interconnected in a hierarchical network by part-of and is-a relations.

The first part of a problem schema comprises knowledge of the problem class and its constituent features, which is important to recognize and activate the appropriate problem schema. In that way, a problem schema serves to better understand the problem. Experts usually have a better understanding of a given problem in their domain in that they can represent the problem based on its second order features rather than its first order features [47]. Second order features seem to be coherent with the structure of the problem, in opposition to first order features, which are coherent with the surface structure. The second order features are the ones that allow deeper elaboration and understanding [9]. Once a problem schema is recognized and triggered, its solution procedure can be applied to the representation of the problem. The solution itself is a structure that has slots, which can be filled by arguments that are taken from the problem representation. As discussed in the previous sections, the slots allow for variations, for ranges of acceptable values. This is important, as it allows to adapt the generic parts of the problem schema to the specific operands given in a problem.

4.3 Subdivisions of problem schemas

Marshall [9] who has researched problem schemas for mathematical problem solving, suggests that a problem schema consists of identification knowledge, elaboration knowledge, planning knowledge and execution knowledge. Thus, she splits the problem and the solution structures into sub structures. Though this split up is plausible, it is rather arbitrary. Does every problem schema require execution knowledge? Think again of the car, which can be the solution to travel from A to B. Once we realize that a car is a proper solution, what is the proper execution knowledge? Do we have to know how to build a car, or only how to drive a car? Is having a driving license a precondition to let the car be a proper solution? What about asking a friend to drive? There are other ways to subdivide problem and solution. In the context of design patterns the problem is usually split up into the core problem, the context, and forces (to be exact: the conflicting forces are the problem and the configuration of forces is set by the context). Each is an interrelated sub structure of the problem structure. Indeed, one can further split up the context as there are several types of contexts (available resources, required skills, cultural habits etc.). The solution, too, has different sub structures, including the form of the solution (a car), the process of creating the solution (design and fabricate the car) and how to use the solution (driving the car). This nature of problem schemas can be summarized as follows:

1. A problem schema is a single schema that relates a problem to a solution.

2. Both, problem and solution are sub schemas that are integrated into the problem schema.

3. Both, problem and solution can be further sub divided into sub schemas.

4. If the problem schema is activated, then it automatically triggers the solution schema as a whole.

5.The solution schema activates its related substructures such as planning, execution, elaboration, use forms etc.

The problem schema is the body, which integrates all the sub structures into one schema. Each of the structures is an abstraction of concrete instances in which the structures have been found. Hence, each structure is a pattern. It is remarkable that in the schema literature, the expression "pattern" is often used only for the identification task, that is the pattern recognition. Of course, the assigned procedures and plans are patterns as well as the they capture the invariants of the solution. The combination of the problem pattern and the solution pattern is captured in a problem schema. Thus, the design patterns of an individual are equivalent to problem schemas since design is considered to be a problem solving task in the pattern community. To be more precise, design patterns are a special case of problem schemas as there are other types of problems.

Both, design patterns and schemas can have relations to other concepts (patterns or schemas). By the combination of several schemas/patterns, composed schemas/patterns result. Therefore, the patterns of problems can be linked to the patterns of solutions. Also linked can be the patterns of contexts, values, consequences, execution knowledge, elaboration knowledge, forces etc. Since each schema defines a configuration space, problem schemas provide a linkage from a problem space to a solution space in the very same way as design patterns intend to do.

Problem schemas apply to knowledge-rich rather than knowledge-lean problem solving [6]. Similar, design patterns apply to design problems that require a large body of information and specialist experience that should be available to the designer [24]. As with design patterns, problem schemas can be utilized in combination to solve larger and complex problems. If a certain combination of problem schema reoccurs, it is likely that this combination establishes a new schema with the participating schemas as variable slots. Which leads us to the question that has been elegantly excluded so far: how are schemas acquired?

5. Schema Acquisition and Development

According to Piaget, schemas are created and adopted by two interplaying processes: assimilation and accommodation. Assimilation is the integration of external elements (perceived stimuli and the data available from the output of other schemas) into developing structures or intro already existing structures. In the assimilation process, properties of the external element that are incoherent with the activated schema will be ignored. However, not every feature can be skipped. Thus, to integrate the current stimulus, the existing structure has to change according to the special properties of external elements. This process is called accommodation. If a schema has assimilated many experienced elements and the accommodation process has established a logical structure that is capable of representing all of them, then the schema becomes stable. It is in a state of equilibration in which it is resistant against outliers and interferences.

5.1 Assimilation

Assimilation is the application of subjective schemas to represent a perceived situation, or perceived objects. Which schemas are appropriate to represent the situation has been discussed in the section Activation of Schemas. Since schemas are abstractions based on previous experience, the empirical information of the currently perceived elements is unlikely to fit completely to the activated schemas. In the assimilation process the information is adopted so that it can be integrated to the activated schemas, that is irrelevant features or superficial variations are ignored. The structure of the perceived object is altered so that it can be represented by schema instantiations. This implies that our perception of things depends on our previous experience and what we expect about situations. For this reason, optical illusions let us perceive things that are not really there. This fitting of perception to expectation does not only apply in everyday situations but also in scientific scholarship, for example new findings are tried to be harmonized with existing mathematical concepts or specific models and theories.

The assimilation process can be expressed by this formula [48]:

$$(T + I) \rightarrow AT + E$$

T is the existing structure, I is the integrated element (the perceived stimulus), E are the eliminated components (the structural parts of the stimulus that are ignored in favor of the schema) and A is a coefficient > 1 that expresses the strengthening and addition to the existing structure. As an example consider the simple schema structures A and B (spatially represented) and the example stimulus in figure 2.

Though the stimulus is not exactly represented by schema A - the last component (variable) has a different position (value) - it matches better with A than with B. Therefore, schema A is activated and assimilates the stimulus. If the stimulus is fully assimilated by A, then its minor difference is ignored. That is it will not be stored permanently in memory; rather, the given stimuli is represented by A. Assimilation is conservative and tries to subordinate the environment into the organism [49]. How exactly the specific experienced stimulus can be retrieved (reproducing assimilation) depends on the specialization of schema A. If A is a very specific schema then it prescribes a detailed structure; hence, it has to be very similar to the perceived stimulus in order to assimilate it (recognizing assimilation).

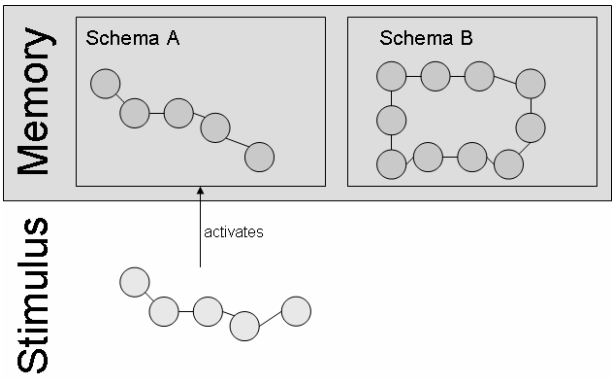

Figure 2: A stimulus is assimilated by the best fitting schema and activates it.

However, if A is on a more abstract level, then it will accept a wider range of stimuli. If the stimulus is assimilated by such an abstract schema, less exact details are available for retrieval since A generalizes over some features (generalizing assimilation). This explains why story fragments that represent typical situations are retrieved by subjects in terms of general information rather than the specific information given in the text. For extraordinary situations, however, subjects often remember the specific details [11].

Besides the differentiation between reproducing, recognizing and generalizing assimilation, Piaget distinguishes between simple and reciprocal assimilation. Simple assimilation just means the integration of external elements into an existing schema. Reciprocal assimilation means that a schema integrates sub schemas (has-a relations), or one schema assimilates another schema and vice versa (is-a relation).

The integration of new data structures without changing the existing schemas is referred to as accretion of knowledge by Rumelhart and Norman [15]. Though they do not speak of assimilation, the proposed process of accretion is very much alike. It is also based on schema activation, or as the “natural side effect of the comprehension process” [15]. In their model, the specific situation or experience is stored in data structures, which are instantiation of appropriate schemas. To retrieve information of a particular experience, the instantiated schemas are used to reconstruct the original experience. This instantiation allows the storage of episodic knowledge while the schema contains the generic structural knowledge. The model differs from Piaget’s assimilation process in that specific data is stored separately in instantiated data structures. Thereby it is better suited than the Piaget model to explain how one can remember specific details that are distinct from the generic structure. However, it does not explain why specific information is lost (forgotten) or replaced by generalized knowledge. The similarity to assimilation is that existing schemas are required to interpret and memorize new input, the new experience is associated with a configuration of schemas, and that the structure of the schemas does not change.

The creation and evolution of schemas is credited to a different process: accommodation [14], or tuning and restructuring [15].

5.2 Accommodation

If there was only assimilation, no learning would occur because there would be no process of changing or restructuring in the knowledge structures. Accommodation is the counterpart of assimilation as it adopts the schemas to be coherent with the new experience. If an external element is assimilated not all of its special features can be discarded or ignored. So, the assimilation schema has to be changed in a way that it can represent the specific new experience as well as the older ones. The requirement that older experiences of the same class must remain representable limits the process of accommodation, however. This limit is expressed by the term A in formula (T + I) -> AT + E . The new schema structures are a cognitive adoption of former schemas.

Analogous to assimilation, Piaget distinguishes between simple and reciprocal accommodation. Simple accommodation happens whenever a schema is activated. Reciprocal accommodation happens if multiple schemas are assimilated by a superior schema, and the coordination of the sub schemas is adopted.

Rumelhart and Norman [15], too, distinguish two types of schema development: tuning is the evolution of existing memory structures (that is of a single schema); restructuring is the creation of new ones (e.g. create a new ensemble of co-operating schemas, or copy an existing schema for modification).

Tuning only affects an adjustment of variables, their values ranges, and probabilities for both values and co-occurrence of values.

The constant and variable terms can be changed in four ways:

1. The accuracy is improved by having more differentiated constraints.

2. The applicability is generalized by extending the range of acceptable variable values.

3. The applicability is specialized by limiting the range of acceptable variable values, in the extreme by replacing it by a constant.

4. Default values can be established if instances of the schema happen to be frequently assigned with typical values. [15]

As an example, consider how a schema accommodates as a number of similar stimuli are assimilated into the same schema (figure 3).

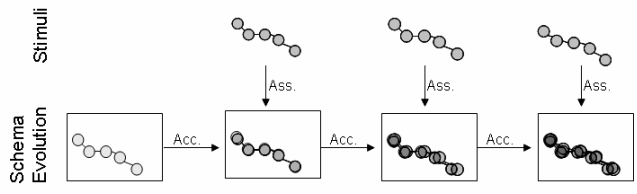

Figure 3: Evolution of a schema by assimilation and accommodation

The number of components (or variables) does not change in the evolution process. However, the ensemble changes in a way that the configuration space extends. The positions of the components (i.e. the variable values) get a wider range. Some configuration values are more probable because they occurred more frequently (in the figure, the strongest memory trail is where the stimuli overlap, thus marking a default but not a mandatory configuration). Also, the relational structure does not change. The schema remains one schema, and its interrelations do not change.

Another type of learning happens if the existing schemas are restructured. In this process new schemas are created either because the new experience does not fit to the currently available schemas, or its organization is not satisfactory. Rumelhart and Norman [15] name two types of schema creation as a result of restructuring: patterned generation and induction.

A patterned generation is a creation of a new schema by copying and modifying an existing one. As in the use of analogies, some variables are left out, others are added and some are changed in their values. For example, one can develop the schema of a rhombus, by getting the information that it has the same relationship to a square that a parallelogram has to a rectangle. Since the old schema is first generalized and then specialized, the logical structure of inheritance is implicitly given. One can say that this type of restructuring concerns is-a relations.

The other form of restructuring is schema induction. As it concerns the contiguity of schemas it will restructure has-a relations. The co-occurrence of certain configurations of schemas will generate a new schema that stores this formation. Though not mentioned explicitly by Rumelhart and Norman [15], one can assume that schemas can not only be created by composition of sub schemas but that a schema can be decomposed into sub schemas as well.

To illustrate the two types of restructuring, consider figure 4 in which the schema evolution goes on by perceiving more stimuli.

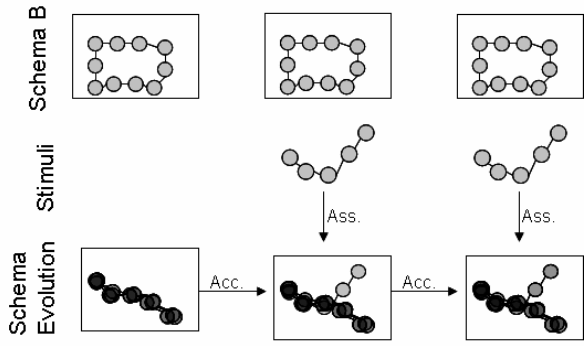

Figure 4: Continued schema evolution

Although the new stimulus is not similar to the evolving schema, it is assimilated by it because it matches this schema better than the competing schema. However, the difference in the last two stimuli require accommodations. The configuration space of the evolved schema is not satisfactory for the experiences it represents. Therefore, restructuring (or reciprocal accommodation in Piaget's terms) is required. The original schema can be copied and modified according to the specializations.

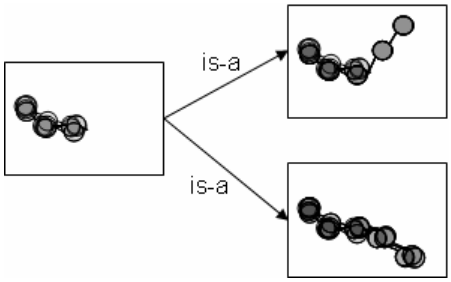

Figure 5: Restructured as patterned generation.

Schema induction, on the other hand, would create new schemas by composition or decomposition. In the example, this would mean, that the schema is decomposed into three sub schemas (figure 6) that can be aggregated into two macro schemas (figure 7).

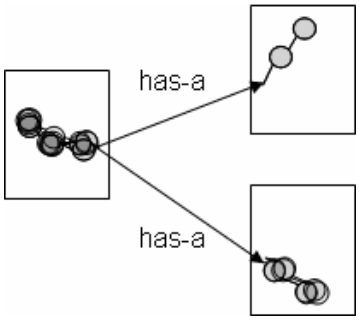

Figure 6: Decomposition into three sub schemas.

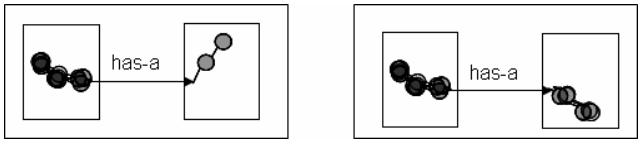

Figure 7: Aggregation into two macro schemas.

In the model of Rumelhart and Norman [15], learning can occur without accommodation. Accretion of knowledge stores the special instances of experience separate from the schemas and therefore does not tune or restructure the schema. In opposition, in the schema model of Piaget assimilation cannot occur without accommodation and vice versa because all knowledge is stored as schemas in the memory. Assimilation always comes first and the assimilation structures compete for activation. Accommodation is subordinated and forced by the fact that the assimilated structures must fit into the configuration space of the schema structures.

5.3 Equilibration

Not every new experience will completely reorganize the existing schemas. For example, if one sees a car with three wheels, the car can be assimilated by the CAR schema. However, the CAR schema will not be accommodated in a way that cars with three wheels are a common pattern. The reason is that with every integrated experience the probabilities for certain configurations are modified. If one has seen thousands of cars with four wheels, the one car with three wheels will not change the schema significantly, if at all. Such a stable schema is in an equilibrated state. The schema structure, that is the variables, their constraints, interrelations and occurrence probabilities, describes a configuration space that is capable to represent all members of a class even the extraordinary ones. Although schemas become stable and equilibrated, this does not mean that there is no change any more. One can assume that a schema is never fully equilibrated because after thousands of hours of practice, an individual can still improve his or her performance [50,51]. The point is that the changes concern the automatization of existing schemas rather than the construction of novel knowledge structures.

Because the schemas are built upon the experiences of an individual, no two persons will have exactly the same schema. Though each person may have a schema of a CAR, these schemas are not identical. They may be very similar but there are also differences. A person that lives in North America may have developed a different prototype of a car in mind than a European. The reason is that people in North America have developed different default values for the variables. Also, the first car one has seen may play and important role since it was the initial ground for the schema. Some people may know about how to check oil and test the air pressures while others lack this information. Some people may be concerned about environmental issues when thinking of cars while others see cars as status symbols rather than transport vehicles. So, even for cars there are many different schemas that may have had developed among individual peoples.

5.4 Related Theories

It is important to keep in mind that schemas are only a conceptual model of human memory; there are no physiological entities in the human brain that correspond to a schema. Rather, schemas are the structural units we can become aware of. More or less, other theoretical approaches such as categories [52], frames [53], scripts [54, 55] or even mental models [56] refer to the same memory structures but offer different explanations how the structural units are stored, related, created and instantiated. Categories are basic concepts (such as ANIMAL, DOG or CAT) whereas schemas can also represent more complex mental structures [57] and situations such as DOGS HUNTING CATS. Scripts are schemas that contain specific sequences of events in well-understood contexts, the classic example is the visit to a RESTAURANT. Mental models, too, can be generated by the interplay of appropriate schemas. What is common in all these approaches is that they provide conceptual models of classification. One can call the class of CARS either a schema, a category, a script, a mental model, or a pattern – depending on the theoretical framework you are in.

Neuroscience goes beyond conceptual models since it tries to map conscious states to activity patterns of firing neurons. The fact that neuroscientists, too, speak of patterns highlights the importance of structural relation and co-occurrence of features in stimulus. "…, human brains operate fundamentally in terms of

pattern recognition rather than of logic. They are highly constructive in settling on given patterns and at the same time are constantly open to error" [58]. Distant neurons will make synaptic connections if their firing patterns are temporally correlated. The synchronized firing creates neural clusters and patterns that correlate with mental states of an individual – the mind emerges [59]. The existing synaptic connections influence the activation patterns that respond to a given stimulus while at the same time the synaptic connections are modified themselves [60]. The conscious mental state is represented by the "integrated pattern of neural activity" [61] and this activity depends on both the current stimuli and the individually developed brain structure formed by former activation patterns. This mixing of the past with the present stimuli to a phenomenal experience is conceptually described as assimilation in schema theory. The altering of the brain structure by strengthening synaptic connections between temporally correlated firing neurons can be considered as accommodation. Then, are the neural activity patterns instantiated schemas and therefore the design patterns as we know them? Not exactly because schemas and design patterns are located on a macro level whereas neurons and synaptic connections are on a micro level in a hierarchical network of activation patterns [62], see figure 8. In the same way the physical object of a car can be decomposed into its components and further into its molecular structure, the schema of a CAR is decomposable into sub schemas and finally into neural activity. In our conscious thinking we have only access to the upper levels. The lower levels can be captured by brain scans. However, today's research is far from linking the recorded brain activity to the actual conscious state. Due to the complexity of dynamic systems this may never be the case. For our enterprise, the mining of design patterns, we are interested in the macro patterns, anyway.

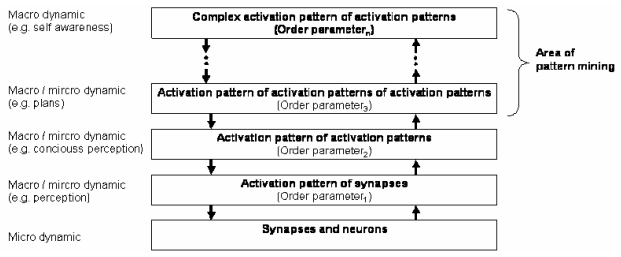

Figure 8: Hierarchy of complex activation patterns [62]. Extended with a note at which levels pattern mining takes place.

6. Implications

At the end of the equilibration process, an individual has developed an ideal of the object in question. And in that, one has developed an attitude towards the ideal object, giving the individual the capability to judge whether the things or events in questions are the way they ought to be – whether they just feel right. This is true for non-design objects (such as trees), for emerging situations and structures (such as sunsets or the behaviour of animals in the woods), and for deliberately designed objects (such as cars, or a software systems) certainly as well. Having an ideal concept of a thing lets one judge the beauty of it. It is therefore, that aesthetics matters [63] and that we should be interested in creating beautiful code [64]and things. From a philosophical perspective, Kant [65] argues that we perceive objects as beautiful when they are good projections of the ideal form. A less philosophical and more pragmatic realization of an ideal form is expressed in the aforementioned statement that is often heard as a judgement for design: "It just feels right!" In other words, things are just the way they ought to be. However, according to schema theory this ideal (or the prototypical instantiation of a schema) is an individually constructed structure and not a-priori given (as Kant has argued). The judgement depends on our experiences and therefore it is grounded in context. In a different time or a different culture the same structure may be judged differently. For this reason we have to reject the hypothesis that there always is a universal beautiful [66] or timeless way [22] of creating and doing things. However, within a culture and a specific period of time it is likely that conventions and socially constructed patterns (the patterns in artefacts of the time) lead to common judgements about what is beautiful or not. While some forms loose their "quality without a name" quickly as fashion comes and goes, there are other forms which offer values across cultures and throughout history more constantly.

What feels right or seems to be ideal depends on the fitness between context and the solution form. Both context and our understanding of the context can change. New technologies or environmental developments can change the context and rearrange the forces. New scientific findings lead to a better understanding of the context and uncover new forces, i.e. not polluting our planet is a force that is often ignored by many industrial designs. To discover that a specific design can do harm can make a once beautiful form look very ugly.

Things that we fully understand (or think to understand) can feel intuitively beautiful. In this case the judgement depends on our implicit design knowledge, stored in equilibrated problem schemas that evolved from the experience of seeing things that worked or not. In other cases, the elegance of design is less obvious and we are required to evaluate the values, benefits and consequences consciously. We have to reflect all the known facts relevant to a design rather than depending on intuition alone [24]. For example, whether a software architecture has sustainable values cannot be judged by only glancing at it. One has to understand the forces and needs the expertise to know which forms balance the forces. At this point the explication and externalization of problem schemas in the form of patterns becomes a helpful tool to reduce the complexity (allowing to reason about more forces), to share information and to ensure that the (mental) models of an individual are also shared by other people.

6.1 Patterns as subjective structures

Patterns depend on schemas constructed by the individual. The more experience an individual has, the more stable the schema of that person get. If one has experienced 100 exemplars, one is more likely to judge based on experience which parts of the configuration are invariant and which change from exemplar to exemplar. Likewise, an expert is aware of commonalities and differences, the expert has many patterns, abstract and precise, in her or his mind. "A man who knows to build has observed hundreds of rooms, and has finally understood the 'secret' of

making a room with beautiful proportions" [67]. This expertise allows the creation of new things by copying good designs. For the domain of architecture, Alexander states [68]: "A pattern language is really nothing more than a precise way of describing someone's experience of building."

The recognition of patterns depends on the cognitive skills of an individual. Even if you systematically analyze design artefacts and define criteria to classify similarities of the objects, these are individual judgements. Formalizing does not work because we are not only interested in a pattern of form but in the pattern of form as a solution; only as such it becomes design and has a meaning. It resolves the forces of a problem in a specific context. There is a difference between the form of a hammer, and the semantic of a hammer as a tool that can be used in certain situations to solve a physical problem. The semantic is something that goes further than the form itself and it requires a human being to make a meaning of the hammer. Therefore, an algorithmic approach must fail. Alexander appreciates that fact when he dispenses from his proposed Program as a method to find pattern diagrams and realizes that the human capabilities allow finding the patterns in a more natural way: "If you understand the need to create independent diagrams, which resolve, or solve, systems of interacting human forces, you will find that you can create, and develop, these diagrams piecemeal, one at a time, in the most natural way, out of your experience of buildings and design, simply by thinking about the forces which occur there and the conflicts between the forces." (preface of 1971 edition of *Notes on the synthesis of form* [24]).

This view is very coherent with schema theory. In the same way that the interaction of assimilation and accommodation leads to an equilibrated schema, the attempts of explication and externalization are evolutionary processes. An implicit schema emerges by experience and to be stable it has to properly represent the object space it refers to. Based on implicit schema, an individual can create an explicit mental representation of its structure in the pattern format. For example, one can implicitly know that a car is a good solution for a problem in various situations. To consider a car in the dimensions of context, problem and solution creates an explicit view on the car as a design pattern. This view will be adapted several times to ensure that it is coherent with the implicit schema it is based on. If a first stable version is available in the mind, one can start the externalisation process by writing the pattern. The written pattern description again will be adapted piecemeal to be coherent with the explicit mental view on the design pattern. Indeed, this process is not a one way path. Writing the pattern will take influence on the mental representation of the design pattern. Reasoning about one's own problem-solving strategies creates new experiences. Thus, the implicit problem schemas may change as well in this process.

6.2 Loss of information (and reality)

The trouble is that a lot of things can go wrong because our minds may be powerful pattern recognition apparatus but they are not perfect. First of all, in any transformation there is loss of information. The richest in-form-ation is available in the formation of a single design artefact itself. Even an objective pattern (if we could grasp it) has to omit a lot of details because it is an abstraction of all the artefacts that manifest the pattern. While the (hypothetical) objective pattern forms a specific design space of all designs of its class, the subjective schema of an individual is only based on a finite subset of this class. This leads to the slightly different ideas of cars and their uses depending on where you live and several other factors. Furthermore, the mental image of a single object (e.g. a car) is always only a projection of the object itself. Likewise, the explicit mental pattern is a projection of the implicit internal schema and the written description is yet another projection. Each projection means loss and sources for failures. While the loss of information can simplify things and make them easier to handle, the danger is to loose critical information, that is to ignore design elements that matter. Failures are even worse because it means that an individual has constructed a mental representation of a real world pattern (or some parts of it) in a wrong way. These are matters of the validity of a design pattern; one could judge the epistemic value of a pattern description by "simply" testing whether the designs of the real world actually manifest the proposed pattern or not. This is a question for the scientists, and an interesting one indeed: How to measure the correctness of a pattern?

One of the pitfalls in evaluating the correctness is individual judge on whether a design is good or bad. The human mind builds up schemas of all kinds of recurrent structures it perceives; including horribly bad designs. The pattern community calls those recurrent designs anti-patterns or dysfunctional patterns. "Some patterns have recurrence but not quality. These are bad patterns" [21]. But dysfunctionality is not always a matter of objective judgement. Everybody agrees that software that can be easily adapted to new requirements is better than software which lacks this capability. But not everyone agrees that PowerPoint adds value to the way people present information. It is a matter of personal preference and the quality of presentations one has experienced in the past. In fact, many of the patterns in "A Pattern Language" are related to values that depend on individual attitudes. For instance, not anybody may agree that a world government, as proposed in the first pattern INDEPENDENT REGIONS [34], is part of a good design. It may be, but people have different judgements on that issue. Such differences are the reason why different parties act in democratic systems.

In the pattern community, people often trust in validity, because it is assumed that the pattern is written by an expert – and in design expertise matters more than (laboratory) experiments. The designer, however, is not satisfied by validity alone. Besides validity the designer expects usefulness and generative qualities from a pattern. A pattern can be scientifically valid by describing the structure correctly and analyzing the consequences – benefits and liabilities – adequately. However, the choice of the level of abstraction, granularity, and details influences the usability for designers. Indeed, some schemas only represent recurrent structure in design but not design patterns since they do not capture adequate solutions to problems. Some people see ideal forms where there are none; this mild form of "platonic schizophrenia" has been anticipated by the pattern community [69].

As an example, let us reconsider the CAR pattern and see what could go wrong with the pattern mining, both informally and systematically. First, one could miss some important components by saying that in essence a car is four tires, a steering wheel, passenger seats, and a body. And indeed this configuration is a recurring structure in proven design solutions (the cars) and

therefore formally a pattern candidate. The trouble is that this four-component pattern is incomplete. It misses important parts such as brakes, engine, fuel tank and many other things. Without brakes, a car is hardly good design. One may implicitly know that a car requires these components but fail to make this explicit. Second, one could claim components as part of the pattern that are actually not. If someone only observes cars with gear sticks, that person may find that a stick is part of the car pattern. But today, cars with sticks are only a variation, there are also automatic cars. The same with trunks. A car without a trunk is still a car, hence trunks are only an option. If the individual observer only saw cars on streets, one could argue that streets are part of the CAR pattern, too. But they are not: cars can exist without streets and streets do not only serve cars. STREETS are an independent pattern that happens to occur often in conjunction with cars using it. Because it is so comfortable to drive on streets, CARS ON STREETS is certainly another pattern. However, in the mind of an individual, cars and streets may be an inseparable unit. For this person, streets are always part of his CAR schema, the pattern in mind. Third, finding an appropriate abstraction level is a challenge. One has to question whether there is a general class of cars, or, each specific car class should be treated as another pattern. For example, although a race car and a family van share many properties, there are significant differences. And what about busses, trucks, or ambulances? These are all different patterns, but they are all specialisations of the CAR pattern, too. Which abstraction level works best depends on the granularity suitable for the target group. If a pattern goes deep into the details, documenting each class as a single pattern is worthwhile. Otherwise one would have to include all the different variations into one long description. On the other hand, if the variation is only small, treating each variation as a new pattern leads to a pattern explosion that fails to serve the original idea of design patterns to communicate expert knowledge in an efficient way.

6.3 The good news: patterns are shared

Finding a pattern at the right level of abstraction, granularity, and detail seems to be very hard, in particular because each individual constructs own patterns in mind. The good news, however, is that these individually constructed patterns do not differ that much when they are communicated. Otherwise an individual would not know what is meant by the word “tree”. Though individuals have different schemas of trees, there is a socially constructed meaning of what a tree is.

In an experiment [70] we tried to find out whether people actually find the same patterns in design. Based on three design patterns of interactive graphics, 14 graphics were generated. The participants were asked to group the graphics by their similarities concerning the interaction form. The graphics had been chosen in a way that they all varied in some features but also shared some features. Hence, one could argue that all 14 graphics manifested the same very abstract design pattern, or that each graphic manifested a different very specific design pattern. The most interesting finding was that the participants found similar patterns. Though different levels of abstractions were chosen, the patterns of medium level were chosen most frequently. Pattern groups with graphics that shared the most significant features (second order features) were found most often. However, some participants were totally misled and grouped the graphics only according to their superficial features (e.g. first order features such as style of illustration). The frequency of a group chosen by the participants correlated with the similarity of second order features. Groups based on first order features were not chosen, or, only found by single participants. Groups, however, that consisted of graphics similar in second features, were chosen by up to 2/3 of all participants. That is, of all the 16384 different grouping options, 2/3 of the participants chose the same group. The most frequent groups, it seems, were the ones in which the commonalities of the graphics contained were most equilibrated. Here equilibration means that the majority of significant features were invariant between the graphics in such a group. Some participants, however, identified groups that were less equilibrated, or were only equilibrated for some features. Those participants were ignoring some features that were judged to be relevant by the majority. Depending on which features were accounted, the participants came to different groupings. That shows that people do develop different schemas based on the same experiences, but it also shows that some schema configurations are more likely to occur than others.

Externalizing a pattern by writing it down, it becomes empirical vulnerable as other people can comment on the pattern and express their level of agreement or disagreement. Descriptions of design patterns are a way to address differences in the constructed patterns of an individual. Writer’s Workshops [71] and shepherding processes [72] are established in the pattern culture to get feedback from other experts; thus, they are a way of reflection on whether the proposed pattern is coherent with the pattern of other persons. The pattern description also helps people who have difficulties in seeing the similarities in experiences to focus on significant features.

6.4 Knowledge transfer

The goal of pattern writing is to share expert’s knowledge. In schema theory it is assumed that knowledge is stored in structural units, which are basically patterns in the mind of an individual. The pattern format therefore seems to be an adequate vehicle to capture these nuggets of wisdom. It is a form of representation closest that is very close to the way knowledge is clustered and discriminated in memory (according to schema theory). Patterns can, therefore, efficiently guide the reader to construct one’s own schemas. It is important to note that, of course, a schema itself cannot be transferred. A pattern description only sketches the schema, but the understanding and capability of applying the pattern must be constructed by each individual. To support this process, illustrative examples are needed. Constructing a stable pattern requires experience. The more exemplars an individual has perceived, the more stable the constructed schema will be. For this reason, examples play an important role in documenting a pattern since they can clarify the invariants and variations of design. Although a pattern explicitly captures the core of design, multiple examples are needed to induce a schema by a learner [73] in order to explore the essential and core features. Alexander points out that each person must have his or her own version of a language in order to make it a living language: “A living language must constantly be re-created in each person’s mind.” [74]. Pattern descriptions may speedup the creation of the schemas but do not replace this process.

6.5 Outlook

Because schemas are actively constructed and depend on the exemplars one has experienced, the schemas that represent the patterns of the real world will be different structures for each individual. The same is true for patterns. That there are different views on the same patterns can be illustrated by pattern descriptions of several authors that refer to the same pattern. The Gang-of-Four patterns [39] have been re-written many times in other books. There are programming language specific descriptions [75], heavily illustrated versions [76], and even patterns for dummies [77].

Besides different views on the same patterns, there are many variations of a pattern, and a pattern can always be split up into sub-patterns or be combined with other patterns to macro patterns. None of these operations does harm to the validity of the resulting patterns, however they may affect the usability. In understanding how patterns are induced by individuals, one can find better strategies of finding appropriate patterns to document. In the competition of schemas, an appropriate level of abstraction and granularity is usually activated: If one sees a car one will usually think of a car and not of a vehicle or a specific car class. So, to describe a car it is probably better to describe it as a CAR and not as a variation of a VEHICLE.

Maybe the most important issue of this paper is to realize that the patterns in real design, the patterns in our heads, and the pattern descriptions are not the same. In spite of the practical nature of patterns, a pattern is only a theory! Something that is made up by our minds. A pattern does not exist as a real thing, because real things can only manifest patterns. Patterns are design spaces and schemas can be mental representations of such design spaces. Pattern descriptions try to grasp the design space of a pattern. But before we can start to describe a pattern we have to reason about it in our mind. The quality of a pattern description depends (besides writing skills) on the validity and adequateness of the patterns in the heads of the contributors.

Seeing patterns as mini-theories, implies that the underlying hypotheses can be true or wrong. There are different methods to provide evidence for proposed patterns. The Writer's Workshop certainly is such a scientific method [78]. Interviews, observation, and experiments might be further sources for evidence. An interesting path for further research would be to defend the epistemic creditability of patterns and consider how the pattern community has established methods and processes to evolve subjective pattern impressions into objective pattern descriptions.

7. Acknowledgement

We are grateful for the help and support of our shepherd, Fujino "Terry" Terunobu. Terry has been great as a shepherd in showing us directions and paths rather than ruling our writing. We felt that by the inspirations and comments he gave to us, we improved the text in ways that would not have been accessible otherwise. We also want to thank all participants of the "Software & People" Writer's Workshop group at the PLoP 2008 in Nashville. Their suggestions were very useful and the discussion was very lively and delightful! This work has become more beautiful through their help. Some of their comments have launched further investigations; the outcomes will be presented in a successive paper.

8. REFERENCES

[1] DeLano, D. E. (1998). Patterns Mining. In Rising, L. (1998). The Pattern Handbook (pp.87-96). Cambridge: Cambridge University Press.

[2] Gabriel, R. P. (2002). Writers' workshops & the work of making things: patterns, poetry (p. 175). Boston: Addison-Wesley.

[3] Vlissides, J. (1997). Patterns: The Top Ten Misconceptions. Object Magazine. March Issue.

[4] Van der Veer, G. C., and Melguizo, M. C. (2002). Mental Models. In J.A. Jacok, and A. Sears (Eds.). The Human-Computer Interaction Handbook: Fundamentals, evolving Technologies and emerging applications (pp. 52-80). Lawrence Erlbaum & Associates.

[5] Cooper, G., and Sweller, J. (1987). Effects of schema acquisition and rule automation on mathematical problem-solving transfer. Journal of Educational Psychology (pp. 347-362). 79. 4.

[6] VanLehn, K. (1989). Problem Solving and Cognitive Skills Acquisition. In Posner, M.I. (Ed..) Foundations of Cognitive Science (pp. 527-579). Cambridge: MIT Press.

[7] Gick, M.L., and Holyoak, K.J. (1983). Schema induction and analogical transfer. Cognitive Psychology (pp. 1-38). 15.

[8] Cummins, D. D. (1992). Role of analogical reasoning in the induction of problem categories. Journal of Experimental Psychology: Learning, Memory, and Cognition (pp. 1103-1124). 18.

[9] Marshall, S. P. (1995). Schemas in problem solving. Cambridge: Cambridge University Press.

[10] Buschmann, F., Henney, K., and Schmidt, D.C. (2007). Pattern-oriented software architecture. Volume 5: On patterns and Pattern Languages (p. 3). West Sussex: John Wiley & Sons.

[11] Bartlett, F. C. (1932). Remembering: An experimental and social study. Cambridge: Cambridge University Press.

[12] Piaget, J. (1952). The Origins of Intelligence in Children. New York: International University Press.

[13] Piaget, J., and Inhelder, B. (1969). The psychology of the child. New York: Basic Books.

[14] Piaget, J. (1971). Biology and Knowledge. Edinburgh: Edinburgh University Press.

[15] Rumelhart, D. E., and Norman, D.A. (1978). Accretion, tuning, and restructuring: Three models of learning. In J.W. Cotton, and R. Klatzky (Eds.). Semantic Factors in Cognition. Hillsdale, NJ: Lawrence Erlbaum Associates.

[16] Sweller, J., van Merriënboer, J. J. G., and Paas, F. G. W. C. (1998). Cognitive architecture and instructional design. Educational Psychology Review, (pp. 251-296). 10.

[17] Hesse, F. W. (1991). Analoges Problemlösen: eine Analyse kognitiver Prozesse beim analogen Problemlösen. Fortschritte der psychologischen Forschung, 8. Weinheim: Psychologie Verlags Union.

[18] Scharlau, I. (2007). Jean Piaget zur Einführung. Hamburg: Junius.

[19] Alexander, C. (1964). Notes on the synthesis of form (p. 117). Cambridge: Harvard University Press.

[20] Alexander, C. (2002). The nature of order, Book 1. The phenomenon of life (p.79). Berkeley, Calif: Center for Environmental Structure.

[21] Buschmann, F., Henney, K., and Schmidt, D.C. (2007). Pattern-oriented software architecture. Volume 5: On patterns and Pattern Languages. West Sussex: John Wiley & Sons.

[22] Alexander, C. (1979). The Timeless Way of Building. New York: Oxford University Press.

[23] Anderson, J. R. (1983). The architecture of cognition. Cambridge: Harvard University Press.

[24] Alexander, C. (1964). Notes on the synthesis of form. Cambridge: Harvard University Press.

[25] Alexander, C. (1964). Notes on the synthesis of form (p.64). Cambridge: Harvard University Press.

[26] Rumelhart, D. E., and Ortony, A. (1977). The representation of knowledge in memory. In R. C. Anderson, J. R. Spiro, and W. E. Montague (Eds). Schooling and the acquisition of knowledge. Hillsdale, NJ: Lawrence Erlbaum Associates.

[27] Buschmann, F., Henney, K., and Schmidt, D.C. (2007). Pattern-oriented software architecture. Volume 5: On patterns and Pattern Languages (p. 76). West Sussex: John Wiley & Sons.

[28] Alexander, C. (1964). Notes on the synthesis of form (p.108). Cambridge: Harvard University Press.

[29] Alexander, C. (1979). The Timeless Way of Building (p. 260). New York: Oxford University Press.

[30] Eckblad, G. (1981). Scheme theory: a conceptual framework for cognitive-motivational processes. London: Academic Press.

[31] Eckblad, G. (1981). Scheme theory: a conceptual framework for cognitive-motivational processes (p.19). London: Academic Press.

[32] Alexander, C. (1979). The Timeless Way of Building (p. 144). New York: Oxford University Press.

[33] Alexander, C. (1979). The Timeless Way of Building (p. 247). New York: Oxford University Press.

[34] Alexander, C., Ishikawa, S., and Silverstein, M. (1977). A pattern language: towns, buildings, construction. New York: Oxford University Press.

[35] Alexander, C. (1964). Notes on the synthesis of form (p.78). Cambridge: Harvard University Press.

[36] Noble, J. (1998). Classifying relationships between object-oriented design patters. Australian Software Engineering Conference (ASWEC), pp. 98-107.

[37] Corfman, R. (1998). An Overview of Patterns. In Rising, L. (1998). The Pattern Handbook (pp.87-96). Cambridge: Cambridge University Press.

[38] Van Duyne, D., Landay, J. A., and Hong, J. I. (2004). The Design of Sites. Boston: Addison-Wesley.

[39] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). Design Patterns: Elements of Reusable Object-Oriented Software. Reading: Addison-Wesley.

[40] Newell, A., and Simon, H. A. (1972). Human Problem Solving. Englewood Cliffs, N.J.: Prentice-Hall.

[41] Duncker, K. (1945). On problem-solving. Psychological Monographs (pp. 1-113). 58 (270).

[42] Ohlsson, S. (1984). Restructuring revisited, II: An information processing theory of restructuring and insight. Scandinavian Journal of Psychology (pp. 117-129). 25.

[43] Simon, H.A. (1962). The architecture of complexity. Proceedings of the American Philosophical Society (pp. 29-39). 74.

[44] Alexander, C. (1964). Notes on the synthesis of form (p. 74). Cambridge: Harvard University Press.

[45] Simon, H..A. (1964) The Science of the Artificial. Cambridge, MA:MIT Press.

[46] Booch, G. (1998). Patterns. In Rising, L. The Pattern Handbook. Cambridge: Cambridge University Press.

[47] Chi, M. T. H., Feltovich, P. J., and Glaser, R. (1981). Categorization and representation of physics problems by experts and novices. Cognitive Science (pp 121-152). 5.

[48] Piaget, J., Fatke, R., and Kober., H. (2003). Meine Theorie der geistigen Entwicklung. Beltz Taschenbuch, 142. Weinheim: Beltz Verlag.

[49] Piaget, J. (1975). Der Aufbau der Wirklichkeit beim Kinde. Stuttgart: Ernst Klett.

[50] Crossman, E. R. F. W. (1959). A Theory of the acquisition of speed-skill. Ergonomics (pp. 153-166). 2.

[51] Fitts, P. M. (1964). Perceptual-motor skill learning. In A. W. Melton (Ed.). Categories of human learning. New York: Academic Press.

[52] Kelly, G. M. (1982). Basic concepts of enriched category theory. London Mathematical Society lecture note series, 64. Cambridge: Cambridge University Press.

[53] Minsky, M. (1977). Frame-system theory. Thinking: Readings in Cognitive Science. Cambridge: Cambridge University Press

[54] Schank, R. C. (1975). The structure of episodes in memory. In Bobrow, D. & Collins, A. (Eds.) Representation and understanding (pp. 237-272). New York: Academic Press.

[55] Schank, R. C. & Abelson, R. (1977). Scripts, plans, goals and understanding. Hillsdale, NJ: Lawrence Erlbaum Associates.

[56] Gentner, D. & Stevens A.L. (1983).Mental models. Hillsdale, NJ: Lawrence Erlbaum Associates

[57] Zimbardo, P. G., Gerrig, R. J., and Graf, R. (2004). Psychologie. Pearson-Studium. München: Pearson.

[58] Edelman, G. M. (2006). Second nature: brain science and human knowledge. New Haven: Yale University Press.

[59] Metzinger, T. (Ed.) (2000). Neural Correlates of Consciousness. Cambridge, MA: MIT Press

[60] Lloyd, D. E. (2004). Radiant cool: a novel theory of consciousness. Cambridge, Mass: MIT Press.

[61] Edelman, G. M. (2006). Second nature: brain science and human knowledge (p.92). New Haven: Yale University Press.

[62] Mainzer, K. (2008). Komplexität. Paderborn: UTB.

[63] Coplien, J. O. (1996). Software Patterns. New York: SIGS Books.

[64] Oram, A., and Wilson, G. (2007). Beautiful code. North Sebastapol, Calif: O'Reilly.

[65] Kant, I., and Erdmann, B. (1880). Immanuel Kant's Kritik der Urtheilskraft. Leipzig: Voss.

[66] Alexander, C. (2002). The nature of order, Book 1. The phenomenon of life. Berkeley, Calif: Center for Environmental Structure.

[67] Alexander, C. (1979). The Timeless Way of Building (p. 222). New York: Oxford University Press.

[68] Alexander, C. (1979). The Timeless Way of Building (p. 207). New York: Oxford University Press.

[69] Marquardt, K. (2004). Diagnosis: Platonic Schizophrenia. In Marquardt, K., Schütz, D. (Eds): Proceedings of the 9th European Conference on Pattern Languages of Programs (pp. 88-108). Konstanz: Universitätsverlag Konstanz.

[70] Kohls, C. and Uttecht, J. G. (in press). Lessons learnt in mining and writing design patterns for educational interactive graphics. Computers in Human Behavior.

[71] Gabriel, R. P. (2002). Writers' workshops & the work of making things: patterns, poetry. Boston: Addison-Wesley.

[72] Harrison, N.B. (2006). The Language of Shepherding. In Manolescu, D., Völter, M., and Noble, J. (Eds.). Pattern Languages of Program Design 5. Boston: Addison-Wesley.

[73] Quilici, J.L., and Mayer, R.E. (1996). Role of examples in how students learn to categorize statistics word problems. Journal of Educational Psychology (p.144-161). 88.

[74] Alexander, C. (1979). The Timeless Way of Building (p. 338). New York: Oxford University Press.

[75] Cooper, J. W. (2000). Java design patterns: a tutorial. Reading, Mass: Addison-Wesley.

[76] Freeman, E., Freeman, E., Sierra, K., and Bates, B. (2004). Head First design patterns. Sebastopol, CA: O'Reilly.

[77] Holzner, S. (2006) Design Patterns For Dummies. Hoboken: Wiley Publishing.

[78] Gabriel, R. P. Writers' Workshops As Scientific Methodology. http://dreamsongs.com/Essays.html (accessed August, 2008)