A Fix for Fixation?

Re-representing and abstracting as creative processes in the design of information systems

Doris Zahner, Stevens Institute of Technology

Jeffrey V. Nickerson, Stevens Institute of Technology

Barbara Tversky, Teachers College, Columbia University

James E. Corter, Teachers College, Columbia University

Jing Ma, Stevens Institute of Technology

Please direct all correspondences to:

Doris Zahner
dzahner@stevens.edu
Stevens Institute of Technology
Castle Point on Hudson
Hoboken, NJ 07030
201.216.8579

A Fix for Fixation?

43 Pages
1 Table
8 Figures

A Fix for Fixation?

Re-representing and abstracting as creative processes in the design of information systems

Abstract

Fixation prevents the associations that are bridges to new designs. The inability to see alternative solutions, or how known solutions can map onto current problems, is a particularly acute problem in the design of software-intensive systems. Here, we study two related ways of liberating fixated thinking: abstracting and re-representing. We find that both techniques can help designers generate original ideas. However, it is also true that both techniques can lead to flawed ideas, because originality and fitness are negatively correlated in participants' solutions. We discuss ways our results might be used to generate reflective design aid that first help generate original ideas and later cull these ideas.

Keywords: abstraction, re-representation, originality, fluency, design of information systems

Introduction

How does thinking happen? At the core, thinking is associationistic (e.g., Anderson, 1993; Rumelhart & McClelland, 1986), one thought leads to another. The paths these thoughts can take are too numerous to count, similarity on aspects too plentiful to list, growing exponentially with each link. Despite the abundance of associations and paths, minds get stuck in ruts, all too often not going in the right direction or not going far enough. One challenge for creativity is encouraging thought to travel in many directions.

Thought may be associationistic and unconstrained, but problems have constraints. Not all paths of thought lead to viable solutions. And unfortunately, the paths that fail to lead to viable solutions are not marked "Dead End." A second challenge to creativity, then, is encouraging thought in viable directions.

These challenges to creativity have long been noted: the first is promoting divergent thinking, the second, convergent. And the ordering of these challenges makes sense, exactly because it is not problem constraints that typically block associative paths of thoughts. One way to be creative is to first generate a broad range of ideas, then check to see if they conform to the constraints of the problem. This is, in fact, how the mind works naturally, both in memory (Rundus & Atkinson, 1970) and in judgment (Sloman, 1996), first a burst of rapid associations, then a slow evaluation of them.

Design problems are one kind of problem solving (e.g., Simon, 1995). Research in problem solving has, sadly, repeatedly demonstrated failures to transfer known solutions to new problems (Gick & Holyoak, 1980, 1983; Novick, 1990; Novick & Hmelo, 1994; Ross, 1989; Ross & Kennedy, 1990). The typical reason for failure to

apply old solutions to solve new problems is failure to see the similarity of the constraints or the deep structure of the problems. The reason for this failure is, in turn, a general property of associationistic thought that it travels most easily along paths with surface similarity, similarity of content domain, rather than similarity along more abstract attributes, such as similarity of structure. Thus, people may understand that sending radiation from many different directions to converge on a tumor can kill the tumor while causing minimal damage to surrounding tissue but nevertheless not apply the principle of convergence to a problem requiring getting enough fire retardant to an oil well to put out a fire when the field of oil wells is too dense to allow large quantities of retardant to be delivered along any one road. Tumors in the body and oil wells in Saudi Arabia are remote domains with few associations between them.

Cuing thinkers in a variety of directions, then, is likely to increase the numbers of thoughts, as well as the kinds of thoughts. For any domain, and for design in particular, the kinds of cues are key. What is the best way to increase the right kinds of thoughts? Some insight comes from close observation of the design process. Designers sketch their ideas, externalizing them (e.g., Schön, 1983). Although externalizing ideas can promote thought by reducing memory load, it can also freeze ideas, resulting in fixation. When students studying information systems were asked to sketch several alternative designs for a single problem, their second alternative and even their third and sometimes fourth alternatives did not stray far from their first (Nickerson et al., 2008). Yet externalizing ideas in sketches also allows contemplating them, reorganizing and restructuring them, and reinterpreting them. Experienced designers know how to reorganize their sketches, to see them differently, and to get new design ideas from that process (Goldschmidt, 1991,

1994; Schön, 1983; Suwa & Tversky, 1997, 2003; Suwa et al., 2001). When designers are advised to perceptually reorganize sketches, they produce more interpretations of the sketches (Suwa & Tversky, 2003). To be productive, perceptual reorganization, a divergent, bottom-up process, must be accompanied by interpretation, a convergent, top-down process (Suwa & Tversky, 2003). Together, these processes effectively serve to re-represent problems.

Expertise helps designers and other problem solvers overcome associations and proposed solutions based primarily on domain specificity and to see and search for underlying similarities of deep structure. But efficiently training this facility has been elusive. One requirement is abstraction, and in fact transfer of ideas from one domain to another can be fostered by presenting problems in an abstract or general form (e.g., Goldstone & Sakamoto, 2003; Goldstone & Son, 2005). However, abstractness is a "Goldilocks" issue. When problems are presented too abstractly, without any domain instantiation, successful solutions decrease (Holyoak & Cheng, 1995; Wason & Johnson-Laird, 1972). Rather, training should be not too abstract and not too specific, but just right. Very abstract problem formulations presumably reduce the number of associative links; some specificity of content is needed to get any associations at all. Here, we apply that reasoning to design, in particular, design of information systems. Given that abstract (but not too abstract) formulations of problems promote transfer, abstract formulations of problems may also promote more diverse domains of solution, one component of creative design. We present design problems in either abstract or concrete forms. In one experiment, participants were students in a course in information systems design, and we assessed their designs for originality and for whether their solutions matched the given

problem constraints. In the second experiment, participants were from a general population, and were asked to generate as many solutions as possible.

In order to measure creativity, we adapted previously proposed models of creativity (Finke, 1990; Maher, 2008) where creativity is measured on two scales. These scales consist of one dimension that is a measure of originality or novelty, and another measure that evaluates practicality, usefulness, or correctness.

Here, we ask designers to take a prototypical information system configuration (e.g. Gamma et al., 1995) and find other situations for which that configuration is appropriate. That is, we ask them to transfer knowledge from one domain to others. Their answers may exhibit fixation, that is, low originality, if they are merely syntactic substitutions of one semantically similar term for another. For example, it lacks creative flair merely to change "Fedex" to "UPS" in a problem involving shipping. Syntactic substitutions will probably be correct but uninteresting, whereas wild transfers across domains may be interesting but incorrect – that is, they may fail to satisfy the constraints of the problem, by mapping a pattern partially or inconsistently. The two creativity scales in this case will measure originality and fit, fit defined as the fulfillment of the constraints of the question. Fluency is sometimes used as a different measure of creativity, and so we also will measure creativity by counting the number of unique ideas generated by a designer (Amabile, 1996; Gasper, 2004; Wallach & Kogan, 1965).

Divergence and convergence are two different processes: in one, we seek many and disparate associations, following the hops of the mind wherever they lead, and in the second, we analyze and prune, combining and eliminating. It is difficult to do both together, and, because the processes are antithetical, we do not expect them to be

correlated. In particular, more original solutions will, in general, be less likely to fulfill constraints. We predict that designers presented with a concrete, specific situation will be less likely to generate original analogs than those presented with more abstract situations. In other words, designers presented with concrete situations will be more likely to fixate, generating uninteresting but usually correct analogs, and designers presented with abstract situations will be more likely to generate original but often incorrect analogs.

In practice, systems designers are usually presented with concrete situations. How can they avoid fixation? Perhaps they can bootstrap. That is, they can work from the concrete situation provided, re-representing it in a more abstract fashion. This term, re-representation, was coined in cognitive psychology to describe the ability that emerges in childhood to represent knowledge at a higher level of abstraction as a consequence of repeated cycles of representing and applying (e.g., Karmiloff-Smith, 1993). It has also been applied to the study of architectural design, in which multiple and varied representations are important (Oxman, 1997). Here, we use it to mean a process of sequentially representing an idea in different mediums, different levels of abstraction, or both. Once something has been re-represented at a higher level of abstraction, it becomes easier to cross domains, generating new ideas and analogs.

If the technique works, then it has potential to help designers create more and better designs, by applying good solutions in one domain to other matching domains. It is not obvious that the technique will work, because, having seen the original concrete situation, it may be that designers will remain fixated. On the other hand, expert designers do re-represent their own work: in architecture, they sketch, and then observe the sketch anew, finding new interpretations (e.g., Goldschmidt, 1991; Schön, 1983;

Suwa & Tversky, 1997; Suwa et al., 2001). It is plausible that expert designers perform a series of sketches in order to abstract from the original situation, thereby encouraging freer association. Thus, we predict that the process of re-representing a concrete situation will also contribute to generating original solutions. It is even possible that the re-representation process itself is more important than the re-representational medium. That is, textual re-representation may work about as well as diagrammatic re-representation, though at some level, diagrammatic and linguistic representations of ideas of are bound to inspire different thoughts and inferences.

Experiment 1

In this study, participants were presented with a set of text scenarios describing potential design problems. The text descriptions varied in abstractness. Participants were asked to create a structurally similar but different text scenario for each problem. The task is an associational one. In practice, designers are presented with situations that are similar to situations they have solved before, and successful designers often recognize the applicability of technology pattern from one domain to another (cf. Bergman et al., 2001; Hamming, 1986).

Method

Participants

Thirty-nine students from a Master's level information systems design course participated in this study during the Fall semester of 2007. Details on the curriculum of the course can be found in Nickerson (2006). Participants had varying levels of

expertise: they ranged from novices to working professionals with many years of experience in systems design.

Procedure

An abstract and a concrete version were created for Problems 1 - 4 (see Table 1). Two test forms were created with five problems each; each form was given to a separate group of participants. Form A presented abstract versions of Problems 1 and 3 and concrete versions of Problems 2 and 4; Form B the reverse. The text for the fifth problem had only a single version. However, for this problem participants were instructed to first re-represent the problem in an abstract form (either textually or diagrammatically), then to create a structurally similar but different text scenario

==========INSERT TABLE 1 HERE ==========

Coding

Two raters rated each solution on two components of creativity: originality and whether the solution fit the constraints of the given problem. Originality was rated on a 1-9 scale, with 1 meaning "extremely unoriginal" and 9 meaning "extremely original". The two raters also rated the dissimilarity of every solution to every other solution created for that problem. A dissimilarity matrix was created from these ratings. A total of ten matrices were created, one for each version (abstract or concrete) for the first four problems, and two matrices for the fifth problem, created by dividing participants' solutions according to whether they created a text- or diagram-based re-representation as an intermediate step. The inter-rater reliability was calculated for each of the matrices; for all problems, $\alpha > .95$. The average rating was used for all analyses.

As a manipulation check, four judges also rated the dissimilarity between the two presented versions of each problem. This was also coded on a scale from 1-9, with 1 being "extremely similar" and 9 being "extremely dissimilar". The average rating was used for all analyses.

The second rated aspect of each solution was whether the structure of the participant solutions matched the structure of the canonical problem text. If a participant's solution implied the same network topology as did the canonical text, it was rated as a match and given a score of 1. If a solution did not imply the same topological structure as the canonical text, it was rated as a miss and given a score of 0. This measure of matching can be seen as a measure of correctness. If the solution is a structural match to the given problem, then students are creating problems that are analogous to the original problem. If the solution is not a structural match to the given problem, the participants do not understand the technology, and thus their solutions, while appearing to be novel, are inappropriate. Two raters rated the solutions for matching the problem constraints, with an inter-rater reliability score of $\alpha = .94$. Any differences were discussed and the consensus score was used for all subsequent analyses.

In addition to the ratings of originality and fitting problem constraints, three independent coders were asked to perform a classification task (cf. Choi & Thompson, 2005). The resulting measure was used as a measure of divergence for the group of solutions to a particular problem. For each version of each problem, the raters sorted the solutions into separate groups based on semantic domains. The number of separate groups for each condition of each problem was counted and each group was assigned a domain name. For example, for the "store and forward" problem, the following solutions,

which are slightly edited to correct grammatical errors, were all sorted into the "communications" domain.

1. I attach my message on a carrier pigeon and let him fly to the desired destination. When the carrier pigeon reaches my contact the person reads the message.
2. Due to a disaster, Julie transmits an email about her health to her family. The email doesn't get there due to failures of communication infrastructure. However the email gets collected by a nearby car which, when it travels near the house of Julie, transmits the email to her home/family.
3. I watch the news on TV every day. My friend doesn't have a TV so when he comes over for lunch, I can share the latest news with him.

The raters then went through the solutions again and assigned domain names at a more specific level to each solution. For example, instead of using "communication" as the overall domain for the three problems above, one of the coders changed the domain name of solution 1 to "carrier pigeon", solution 2 became "ad hoc network", and solution 3 became "physical communication". The number of more specific domains each coder created for each condition of each problem was averaged and this average was used for the data analyses.

Results

Number of Domains

According to the classification criteria task (Choi & Thompson, 2005), creativity is greater for those groups that have higher degree of fluency (number of non-redundant ideas) and flexibility (number of domains represented in the ideas). We predicted that the participants in the more abstract conditions would have greater levels of fluency and flexibility, i.e. more non-redundant, unique domains. For all four problems, on average, the coders created a greater number of unique domains at both the general and the specific levels for the abstract conditions than for the concrete conditions.

Originality of Solutions

We predicted that the participant solutions created from the abstract conditions of the problems would be more original than those created from the concrete condition. In order to test this, we calculated the mean (and standard deviation) originality score for each variant of each problem using the scores from our dissimilarity matrix. The mean score for each set of problems was then compared between the abstract and concrete conditions. Figure 1 shows the results of this analysis. For all problems, with the exception of the "store and forward" problem, the originality scores for the abstract condition solutions were higher than those for the concrete condition. A split-plot factorial ANOVA was run on the data and results show that the solutions created under the abstract condition ($M = 3.45$, $s = .62$) were significantly more original than those created under the concrete condition ($M = 2.59$, $s = .65$) across all four problems, $F(1,34) = 4.23$; $p < .05$.

==========INSERT FIGURE 1 HERE ==========

The "store and forward" problem was the only one that did not appear to fit this pattern of greater originality for the abstract version. We posited that this was because the description of the "store and forward" problem was qualitatively different from the other three. In order to check this idea, four different raters rated the dissimilarity, on a scale from 1 to 9, between the two versions of the canonical problem texts. Results showed that the raters judged the "store and forward" problem to have the largest dissimilarity between the two presented problem versions. The "publish and subscribe" problems were rated as having a dissimilarity score of 3.5 between the abstract and concrete versions. The "consolidated database" problems were rated as having a

dissimilarity score of 3.75. The "database lock and unlock" problems were rated rated as having a dissimilarity score of 5.25, as having dissimilarity score of 4.25. However, the "store and forward" problems were which is a whole point higher than the "database lock and unlock" problem.

Upon further inspection of the problem texts, we noticed that the "store and forward" problem was the only one that had only *people* in the concrete condition as opposed to machines. Herein lies a possible explanation. People are so different from machines that the participants creating technical solutions from a social scenario are, of necessity, creating solutions that are seen to be more original. Indeed, of the nineteen solutions that were created from the concrete condition of this problem, fifteen of them involved electronic connections, even though the problem itself was about social connections. As a result, the solutions for this problem tended to be transfers to different domains, and were considered to be original. The participants who created an alternate solution for the concrete condition for the other three problems did not necessarily change domains since the original problem was also an information systems problem. For these problems, their solutions were considered to be less original than the solutions in the abstract condition. This reasoning suggests that the advantage for the Concrete condition for this problem may have been an artifact of using a social domain example. However, it is possible that to prevent fixation, there may be some value in presenting technological systems through social metaphors: students will be more likely to experience the process of transferring across domains, thereby generating more original solutions. Future research might specifically compare social versus technical instantiations of information systems design patterns to discover their relative merits.

Fitting the Problem Constraints

For this study, we measured creativity on two scales, originality and fit. We predicted that the originality of a solution would not necessarily mean that the solution fits the problem constraints. For example, there are cases where highly original solutions do not fit the problem constraints and cases where solutions that fit the problem constraints are not very original. We compared the participant solutions to the given problem text and coded whether or not the solution fit the problem constraints. Those solutions that fit the problem constraints were considered to be a match and those that did not were considered to be a miss. Figure 2 shows the proportion of solutions that fit problem constraints by problem and condition. As we see in this graph, the concrete condition yielded more solutions that fit the problem constraints than the abstract condition, with the exception of the "store and forward" problem. This result is the exact complement of the results from our originality analysis (Figure 1), indicating that highly original solutions are more likely to be produced in the abstract condition, but solutions that fit the problem constraints are more likely to be produced in the concrete condition. As predicted, the correlation between originality of the solution and fitting the problem constraint is $r(147) = -.44$, $p < .001$, meaning that the more original the solution, the less likely it is to be correct.

==========INSERT FIGURE 2 HERE ==========

In addition to these analyses conducted on all solutions, we investigated the originality score for only those solutions that fit the problem constraint. Figure 3 shows that for two of the four problems ("publish & subscribe" and "lock & unlock"), the

abstract condition yielded correct solutions that were more original than those in the concrete condition. These are the solutions that are most interesting, and the ones that we want to encourage when we train designers. Although there were more correct solutions in the concrete condition, those solutions were often generated by trivial syntactic substitutions whereas the correct solutions in the abstract condition are both highly original and fit the problem constraints.

==========INSERT FIGURE 3 HERE ==========

Re-representing: The "tracking" problem

The fifth problem was different from the other four problems in that all participants saw the same text: *When I want a package to get to a destination quickly and safely I use Fedex, because they guarantee next day delivery, and they allow me to track the package and know that it has been delivered*. We call this problem the "tracking" problem.

After seeing this text, half of the participants were asked to generate an abstract text that fit the problem constraints and the other half were asked to generate an abstract diagram of the problem constraints. All participants were then asked to generate a specific textual example that fit the problem constraints of the original problem.

For the "tracking" problem, we found that there was no significant difference between solutions created by the two groups for originality, indicating that the interim step of using either text or diagram did not affect the originality of the final scenario that is created. However, we predicted that the interim step of re-representing the canonical text would induce participants to generate solutions that were at least as original as the

abstract conditions and significantly more original than the solutions from the concrete condition. Indeed, as shown in Figure 4, we found that the mean originality score for the re-represented problem ($M = 4.45$, $s = 1.72$) was significantly higher than both the abstract solutions ($M = 3.45$, $s = .62$), $t(105) = 3.96$, $p < .001$, and the concrete solutions ($M = 2.59$, $s = .65$), $t(104) = 8.68$, $p < .001$. Therefore, re-representing works at least as well as presenting an abstraction at the start. In fact, it works significantly better for generating more original solutions.

==========INSERT FIGURE 4 HERE ==========

This result suggests that asking students to abstract from concrete examples first, and then generate analogs, is a good way of encouraging originality. It doesn't seem to matter whether the abstraction happens with text or with a diagram. Asking people to re-represent the problem in either medium as they abstract the situation yields more original solutions.

The proportion of correct solutions was also calculated for the "tracking" problem and compared to the previous four problems. As seen in Figure 5, we again observe the phenomenon that highly original solutions are often incorrect, shown by the fact that there is again a negative correlation ($r(32) = -.61$, $p < .001$). The proportion of correct solutions from the tracking problem is lower than the other four problems, although interestingly, the diagram condition yielded slightly more correct solutions than the text condition.

==========INSERT FIGURE 5 HERE ==========

Qualitative Analysis

In addition to the quantitative analyses reported above regarding originality and fit, we also looked at individual responses to see if there was a qualitative difference between solutions that were judged to have high and low originality scores or that fit or did not fit the problem constraints. Here too we found that solutions from the concrete conditions tended to be less original than those from the abstract conditions, but they did tend to fit the problem constraints. Less original solutions tended to be the ones that remained in a similar, if not the same domain as the given text. For example, the canonical text for the "publish and subscribe" problem read as:

> *At Goldman Sachs, traders subscribe to stock quotes they are interested in, and then receive market information for the subscribed companies in real time.*

Many of the solutions tended to be very similar in nature, usually relating to an online subscription to an electronic information feed. For example:

- At A&P, [a] store manager subscribes to sales report they are interested in and then receive[s] the information for the subscribed retail store every day.
- A woman subscribes to a cosmetic shop. She gets updated information about the new products.

There were two instances where participants created solutions that were virtually identical to the canonical text, showing only minor syntactic differences:

- Traders request stock quotes they are interested in to Goldman Sachs. Goldman Sachs requests market information to the requested company in real time. Goldman Sachs send[s] everything to the traders.
- At Goldman Sachs, investors subscribe [to] the stock product information they are interested in, and then receive the product information for the products in real time.

Yet in all cases, these solutions were considered to fit the problem constraints because they matched the structure of the original text.

In comparison to the concrete condition, the highly original abstract condition solutions did not always fit the problem constraints. For example, the following five solutions all receive high scores for originality due to the domains of the solutions. However, none of the solutions fit the original problem constraints of "publish and subscribe."

- I ordered a database management book from Barnes & Noble bookstore through their website and after 2 days, I went and collected the book from their store near my house in Edgewater.

- A person orders "Management Engineering" book online on Borders website and receives the book after a week to his residence address.

- A user updates their address on the company directory in one online application and the data is replicated to another application within minutes (company intranet)

- Many graduate students are able to provide their advisor with their graduate application and candidacy without meeting with the advisor.

- A person orders a coffee at Starbucks [and] when the coffee is ready, the cashier gives it to the person.

There were, of course, solutions that were created in the abstract condition that were both highly original and fit the problem constraint. For example, for the "lock and unlock" problem, the canonical text for the abstract problem read as:

Whenever a request is made to update a certain type of record, the database table is locked, and only unlocked upon the completion of the update.

For this problem, we considered highly original solutions to be ones that were in different domains, such as ones that did not involve databases or computers at all.

- When I use the restroom, I lock behind me and then unlock when I finished.

- It is like a nuclear reactor. Whenever the security person uses the system in nuclear reactor the room gets locked. When he is complete room gets unlocked again.

- When a client comes to a reception desk, the receptionist focuses on the request from the client. The receptionist should not change his/her focus until the client is satisfied and leaves the desk.

All of these examples were considered to be highly original and also matched the problem constraints.

An informal class discussion following the experiment yielded interesting insight into the participants' design processes. The participants claimed that they noticed a difference in abstractness in the various problems. In general, participants found it easier to create solutions from an abstract example. Some reported difficulty creating solutions when given a specific example because they felt that it focused them too much. They became "stuck" in a particular domain. Finally, they felt that going from a specific example to an abstract representation in the "tracking" problem was the most difficult part of the problem set. However, once the abstract solution was developed, it was relatively easy to go back and develop another specific example in a differing domain.

Experiment 2

Experiment 2 was designed to test if the results of the first experiment could be replicated using a different method of measuring creativity. We also wanted to investigate a different population from students in an information systems design classroom. In this second experiment, we chose to use a classic measure of creativity, fluency, (Amabile, 1996; Gasper, 2004; Wallach & Kogan, 1965) and used a much more diverse set of participants who were recruited from an online forum.

In Experiment 2 we asked participants to brainstorm and come up with as many unique solutions that are similar to the given problem. We predict that the participants who saw the more abstract version of the problem will create more unique solutions than those who saw the concrete version of the problem. We also predict that fewer solutions from the abstract conditions will match the problem constraints, as seen in the first study.

Method

Participants

Participants were solicited through a posting on a public website asking them to "Brainstorm: Be Creative! (knowledge of information systems is helpful)." Once they agreed to participate in the study, they were presented with instructions that told them to "Tell me as many situations that you can think of that are similar to the scenario below" followed by one of eight scenarios. Each subject participated in only one of the eight scenarios, with approximately thirty subjects for each question. Two hundred fifty-six subjects participated in this study, 101 females and 155 males and were compensated with a nominal stipend. Their ages ranged from 17 to 69 with an average age of 31. They spent an average of three minutes and twelve seconds completing the task and a related demographic survey. Eighty-one percent were primarily English speakers and 72% had college degrees. Twenty percent had programming experience (10,000+ lines of code) and 13% had more than 5 years of work experience.

Materials

The two versions of the four problems with from the first study (Table 1) were used for this second study, with one change. Because the "store and forward" problem

yielded anomalous results in the first study, which we believed was due to the social domain, we changed the concrete version of the "store and forward" problem to "I email my assistant my expense report and ask her to print it out and hand it to the controller for approval."

Coding

Because each participant was asked to create as many similar situations as they could think of for each of the scenarios, we counted the number of unique solutions each participant created and treated this as a measure of fluency (Amabile, 1996; Gasper, 2004; Wallach & Kogan, 1965). We also coded the solutions, as we did in the first experiment, for whether they matched the problem constraints. Since the inter-rater reliability for fit was over .90 for the first experiment, we only used one of the two original coders to rate solution fit for this experiment.

Results

Fluency

We predicted that the participants would create more alternative solutions for the abstract versions than for the concrete versions of the problems. In order to test this, we calculated the mean (and standard deviation) of the number of solutions created for each variant of each problem. The mean score of the abstract and concrete conditions for the four problems were then compared. Figure 6 shows the results of this analysis. An independent groups t-test was conducted. The results showed that the participants in the abstract condition ($M = 2.45$, $s = .43$) created significantly more solutions than those in the concrete condition ($M = 1.88$, $s = .29$) across all four problems, $F(1, 254) = 9.49$, $p < .01$).

==========INSERT FIGURE 6 HERE ==========

As a result of changing the "store and forward" problem from a social domain to a technological one, the difference in the number of solutions for the abstract condition match the results of the main effect, which was not observed in the previous experiment. However, in this experiment the results from the "consolidated database" problem did not match the results of the main effect. We believe that this could be attributed to the fact that we are employing a different measure of creativity. We also believe that it is because the abstract version of this problem is extremely specific, making it difficult for participants to generate many different solutions. One possible explanation is that problem abstractness is subject to a "Goldilocks" effect: abstract problem descriptions have to be just right in order to promote fluency.

Fitting the Problem Constraints

In addition to counting the number of solutions, we also coded for whether the solutions that were created matched the problem constraints. We predicted that the number of solutions would correlate negatively with the fit of solutions. Figure 7 shows the proportion of solutions that fit the problem constraints. When we compare the results from Figures 6 and 7, we see that fluency is inversely related to fit. On average, 75.37% of the solutions created in the concrete conditions were ones that fit the problem constraints, compared to 57.79% of the solutions from the abstract conditions. This is yet another indication that abstraction may promote divergence, but not convergence.

==========INSERT FIGURE 7 HERE ==========

In addition to the analyses based on all solutions, we investigated the number of correct solutions for the eight problems. Figure 8 shows that with the exception of the

"store and forward" problem, the participants created more solutions that fit the problem constraints in the concrete condition than the abstract condition. However, this does not necessarily indicate that these solutions were better. This merely indicates that abstraction promotes divergent thinking that is not always appropriate for a given problem. Concrete problems yield more solutions that fit the problem constraints, but the more numerous solutions may just be trivial substitutions of the original text. In fact, the overall correlation between the number of solutions created and the number of correct solutions created is $r(252) = .679$, $p < .001$. However, when looking at the correlation between the number of solutions created to the number of correct solutions for the abstraction versus concrete conditions, we find that more correct solutions tend to be made for the concrete condition ($r(126) = .815$, $p < .001$) than for the abstract condition ($r(126) = .632$, $p < .001$).

==========INSERT FIGURE 8 HERE ==========

Discussion

Generating original solutions is a key challenge for designers. All too often, designers fixate on previous solutions. Previous work suggests that abstraction is one route out of the rut. Here, we tried two ways to promote new ideas by encouraging abstraction. In Experiment 1, students in an information systems design class were presented with a design task that required them to generate novel solutions from given cases that were either abstract or concrete in form. In Experiment 2, participants from varying backgrounds generated as many similarly structured scenarios as possible from the one with which they were presented, either abstract or concrete. We investigated

whether the abstractness of the presented cases would influence the originality, fluency, and structural fit of the participants' designs. Both experiments showed that abstractness does promote original ideas in the design of information systems. In the first study, solutions that were created from the abstract condition tended to be significantly less similar in domain and content than those created from the concrete condition, which is consistent with results from a previous study (Tversky et al., 2008). In the second study, more solutions were generated from the abstract than the concrete examples. The experiments used several measures of creativity (originality, structural fit, and fluency) and related creativity to abstraction.

Our results show that creativity and abstraction are associated in the following way. Participant solutions that were considered highly original tended to come from the abstract conditions, whereas participant solutions that were considered to be unoriginal tended to come from the concrete conditions. In addition, participants provided with the abstract examples created solutions with a wider range of domains and created more solutions in general (Figure 6). Therefore, divergence increases with abstraction of the problem. Note that the abstract examples used here were not too abstract; that is, they were not devoid of content. Purely abstract versions, those with only mathematical or logical structure and content, have not worked well in other domains of problem solving (e.g., Johnson-Laird, 1983).

Creating more numerous solutions does have a cost. A smaller proportion of them actually fit the problem constraints (Figures 2 and 7). Highly original solutions were not necessarily better solutions; that is, they did not necessarily fit the problem constraints. This suggests there is wisdom in the prescription common in the training of

brainstorming and other creative processes: to generate first, as widely as possible, and evaluate later (Osborn, 1963). Abstraction and other divergent thinking processes may help generate original ideas, but not all ideas will be useful. Evaluation as a second stage can weed out the inappropriate ideas. One challenge is to find ways to create abstract formulations that make the constraints clear. Such formulations may do double-duty: they may induce generation of a broader range of creative solutions and at the same time to induce generation of solutions that conform to the problem constraints.

In the second part of the first study, participants were encouraged to create an abstract solution by themselves, rather than being provided one, a process termed re-representation. Re-representation was additionally effective in increasing number of proposed solutions. This suggests that a good and quite general procedure for increasing design creativity is to instruct designers to first create an abstract solution before creating concrete solutions When participants were given a concrete case and asked to create an interim abstract case, either in text or diagram form, the final concrete cases that were created were more original than those of abstract and concrete conditions from the other problems. This finding is reminiscent of findings in transfer of problem solution, where successful far transfer was facilitated both by multiple examples and by generating an abstract rule, expressed either as a diagram or a sentence (Gick & Holyoak, 1980; 1983). Creating an abstract case also seems to encourage developing deeper understanding, or mental model, of the situation (e.g., Gentner & Wolff, 2000; Gentner & Stevens, 1983; Kotovsky & Gentner, 1996; Ross 1989). Once a person has an adequate mental model, it is easier to see structural similarities (Gentner & Markman, 2006) between different

situations. This structure mapping promotes transfer, and, in the field of design, may lead to more solutions.

The process of re-representation has another merit: it is a method to reduce fixation that can be practiced by individual designers. In most practical design situations, present problems and past solutions will all be experienced as concrete situations. By consciously abstracting, a designer may free up associational processes.

Ideally, we want designers to generate solutions that are both original and appropriate. It is not clear if this can be done in one step, or whether a two-step process is necessary: generating followed by evaluating and eliminating. In actual practice, these processes are often intermixed; ideas are evaluated as they are generated, and the evaluation can lead both to altering solutions in order to fit design constraints and to new ideas. A good abstraction will conform to problem constraints as well as increase the range of associations and domains. Future research might experiment with manipulating the stage at which evaluation is introduced. This might be accomplished by comparing an outer loop process, in which all ideas are generated and then all ideas are compared, with an inner loop process, in which each idea is checked as it is generated. Thus, generating abstractions as part of re-representation is promising for accomplishing both goals: increasing the originality of solution ideas and assuring that they are viable solutions.

The finding that more abstract formulations of design problems encourage more and more original solutions raises special problems for the practice of information system design. Information system design is typically done for detailed, concrete, often real examples. Concrete elicitation of requirements, as in scenario building (cf. Booch et al., 1998), may be important for many reasons, but such concrete requirements may fixate

designers. The finding suggests that adding a design process of abstracting requirements into patterns outside of any particular domain may be useful, as it may broaden processes of association that could allow the new problems to be mapped to existing solutions (cf. Bergman et al., 2001, 2002).

For design viewed more generally, the results suggest that the design of reflective aids (cf. Redmiles & Nakakoji, 2004) might profitably be focused on ways of encouraging abstraction. One way to do this would be to adopt a fading procedure, a technique that has promoted transfer in systems science pedagogy (Goldstone & Son, 2005). Fading could be instituted by selectively removing labels from sequence diagrams until the domain associations have been reduced. The convergent thinking component – the checking of structural analogs – might also be supported by design aids. For example, automatic analogical mapping systems (e.g. Holyoak & Thagard, 2002) might take the initial problem and the generated analog, and propose a mapping to the user. If the system correctly finds a true mapping, or correctly fails to find a mapping, this will serve the designer by validating or refuting a design alternative. Either way, the outcome should stimulate appropriately focused design thinking.

In Sum

Software-intensive systems are complex artifacts, notoriously difficult to construct, and little is known about how to design them (cf. Lee, 2000). To find solutions, designers think to follow associations to generate possible solutions and attempt to map known solutions to new problems (cf. Bergman et al., 2002). But fixation is a common block in many creative processes. The present research has shown that more abstract formulations of problems free designers from fixation, leading to more original

solutions than concrete representations. However, designers often are charged with designing specific, concrete examples. Here, we tested a technique that, following standard practice, first provides designers with concrete examples. Then, parting from traditions, we asked designers to generate an abstraction, either textual or diagrammatic, before generating analogs to the original examples, a process of re-representation. This procedure generated a greater diversity of solutions than from the typical concrete scenarios. Surprisingly re-representing generated a greater diversity of solutions than an initial abstract scenario. Re-representing appears to be a powerful technique in generating novel solutions.

However, novel solutions do not necessarily conform to the constraints of the problem: consistently throughout these studies we found a negative correlation between originality and fit. Abstraction and re-representation, then, are important ways of generating novelty, but they also generates errors. As for other techniques of divergent thinking, abstraction and re-representation need to be followed by a later editing to remove insufficient solutions.

Re-representing could be augmented through the use of reflective design aids. One possibility is to establish a process of gradual abstraction through, for example, removing labels on diagrams. Another aid would encourage designers to re-represent their ideas through successive abstractions. Yet another would be a tool to support the verification of generated ideas through a dialog with the user.

Truly creative ideas are rare commodities, but truly creative ideas have a high impact. Creating situations that expand numbers and originality of design solutions has a cost, many ideas must be rejected. Nevertheless, the payoff is probably worth it.

Acknowledgements

We are grateful to Yin Jin Rho and Mike McGahan for assistance on various aspects of the project. The work was supported in part by NSF IIS-0725223, NSF REC-0440103, and the Stanford Regional Visualization and Analysis Center.

References

Amabile, T.M. (1996). *Creativity in Context.* Boulder, CO: Westview Press.

Anderson, J.A. (1993). *Rules of the Mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Bergman, M., King, J., & Lyytinen, K. (2001). Large scale requirements analysis as heterogeneous engineering. *Scandinavian Journal of Information Systems*, 12(1), 37-55.

Bergman, M., King, J., & Lyytinen, K. (2002). Large scale requirements analysis revisited: The need for understanding the political ecology of requirements engineering. *Requirements Engineering Journal*, 7(3), 152-171.

Booch, G., Jacobson I., & Rumbaugh, J. (1998). *The Unified Modelling Language User Guide*. Reading, MA: Addison-Wesley.

Choi, H. & Thompson, L. (2005). Old wine in a new bottle: Impact of membership change on group creativity. *Organizational Behavior and Human Decision Processes*, 98(2), 121-132.

Finke, R. (1990). *Creative Imagery: Discoveries and Inventions in Visualization.* Hillsdale, NJ: Lawrence Erlbaum.

Gamma, E., Helm, R., Johnson R., & Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Upper Saddle River, NJ: Addison-Wesley.

Gasper, K. (2004). Do you see what I see? Affect and visual information processing. *Cognition and Emotion*, 18(3), 405-421.

Gentner, D. & Markman, A.B. (2006). Defining structural similarity. *The Journal of Cognitive Science*, 6(1), 1-20.

Gentner, D. & Stevens, A.L. (Eds.). (1983). *Mental Models*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Gentner, D., & Wolff, P. (2000). Metaphor and knowledge change. In *Cognitive Dynamics: Conceptual and Representational Change in Humans and Machines*. (Dietrich, E. & Markman, A.B., Eds.), pp. 294-342. Mahwah, NJ: Lawrence

Gick, M.L. & Holyoak, K.J. (1980). Analogical problem solving. *Cognitive Psychology*, 12(3), 306-355.

Gick, M.L. & Holyoak, K.J. (1983). Schema induction and analogical transfer. *Cognitive Psychology*, 15(1), 1-39.

Goldschmidt, G. (1991). The dialectics of sketching. *Creativity Research Journal*, 4(2), 123-143.

Goldschmidt, G. (1994). On visual design thinking: The vis kids of architecture. *Design Studi*es, 15(2), 158-174.

Goldstone, R.L. & Sakamoto, Y. (2003). The transfer of abstract principles governing complex adaptive systems. *Cognitive Psychology*, 46(4), 414-466.

Goldstone, R.L. & Son, J. Y. (2005). The transfer of scientific principles using concrete and idealized simulations. *The Journal of the Learning Sciences*, 14(1), 69-110.

Hamming, R.W. (1986). You and your research. *Bell Communication Research Colloquium Seminar.*

Holyoak, K.J., & Cheng, P.W. (1995). Pragmatic reasoning with a point of view. *Thinking & Reasoning*, 1(4), 289-313.

Holyoak, K.J. & Thagard, P. (2002). Analogical mapping by constraint satisfaction. In *Cognitive Modeling*, (Polk, T.A. & Seifert, C.M., Eds.), pp. 849-909. Cambridge, MA: MIT Press.

Johnson-Laird, P.N. (1983). *Mental Models: Toward a Cognitive Science of Language, Inference, and Consciousness*. Cambridge, MA: Harvard University Press.

Karmiloff-Smith, A. (1993). Constraints on representational change: evidence from children's drawing. *Cognition,* 34(1), 57-83.

Kotovsky, L. & Gentner, D. (1996). Comparison and categorization in the development of relational similarity. *Child Development*, 67(6), 2797-2822.

Lee, A. (2000). Systems thinking, design science and paradigms: Heeding three lessons from the past to resolve three dilemmas in the present to direct a trajectory for future research in the information systems field. *11th International Conference on Information Management*. http://www.people.vcu.edu/~aslee/ICIM-keynote-2000.

Maher, M.L. (2008). Keynote Address. *Design, Computing and Cognition*.

Nickerson, J.V. (2006). Teaching the integration of information systems technologies. *IEEE Transactions on Education*, 49(2), 271-277.

Nickerson, J.V., Corter, J.E., Tversky, B., Zahner, D., & Rho, Y. (2008). The spatial nature of thought: Understanding information systems design through diagrams. *ICIS 2008 Proceedings,* Paper 216. http://aisel.aisnet.org/icis2008/216

Novick, L. (1990). Representational transfer in problem solving. *Psychological Science*, 1(2), 128-132.

Novick, L. & Hmelo, C. (1994). Transferring symbolic representations across nonisomorphic problems. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 20(6), 1296-1321.

Osborn, A.F. (1963) *Applied Imagination: Principles and Procedures of Creative Problem Solving* (Third Edition). New York: Charles Scribner's Sons.

Redmiles, D. & Nakakoji, K. (2004). Supporting reflective practitioners. *Proceedings of the Ninth International Conference on Software Engineering*, pp. 688-690.

Ross, B.H. (1989). Distinguishing types of superficial similarities: Different effects on the access and use of earlier problems. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15(4), 456-468.

Ross, B.H. & Kennedy, P.T. (1990). Generalizing from the use of earlier examples in problem solving. *Journal of Experimental Psychology: Learning, Memory, & Cognition*, 16, 42 – 55.

Rumelhart, D.E., & McClelland, J.L. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition.* Vol. I. Cambridge, MA: MIT Press.

Rundus, D. & Atkinson, R.C. (1970). Rehearsal processes in free recall: A procedure for direct observation. *Journal of Verbal Learning and Verbal Behavior*, 9(2), 99-105.

Schön, D.A. (1983). *The Reflective Practitioner: How Professionals Think in Action*, Jackson, TN: Basic Books.

Simon, H.A. (1995). Problem forming, problem finding and problem solving in design. In *Design & Systems* (Collen, A. & Gasparski, W., Eds.), pp. 245-257. Edison, NJ: Transaction Publishers.

Sloman, S.A. (1996). The empirical case for two systems of reasoning. *Psychological Bulletin*, 119(1), 3-22.

Suwa, M. & Tversky, B. (1997). What do architects and students perceive in their design sketches? A protocol analysis. *Design Studies,* 18(4), 385-403.

Suwa, M. & Tversky, B. (2003). Constructive perception: A skill for coordinating perception and conception. In *Proceedings of the 25th Annual Conference of the Cognitive Science Society*, (Alterman, R. & Kirsh, D., Eds.), pp. 1140-1145, Austin, TX: Cognitive Science Society.

Suwa, M., Tversky, B., Gero, J.S., & Purcell, T. (2001). Seeing into sketches: Regrouping parts encourages new interpretations, In *Visual and Spatial Reasoning in Design II* (Gero, J.S., Tversky, B., & Purcell, T., Eds.), pp. 207-219, Sydney: Key Centre of Design Computing and Cognition.

Tversky, B., Corter, J.E., Nickerson, J.V., Zahner, D., & Rho. Y. (2008). Transforming descriptions and diagrams to sketches in information systems design. In, Diagrams *2008* (Stapleton, G., Howse, J., & Lee, J., Eds.), pp. 242-256, Berlin: Springer-Verlag.

Wallach, M.A, & Kogan, N. (1965). *Modes of Thinking in Young Children: A Study of the Creativity Intelligence Distinction.* New York: Holt, Rinehart & Winston.

Wason, P. & Johnson-Laird, P. (1972). *Psychology of Reasoning: Structure and Content*. Cambridge, MA: Harvard University Press.

TABLES

Table 1: The four different problem topics with their concrete/abstract variants.

Topic	Concrete	Abstract
publish & subscribe	At Goldman Sachs, traders subscribe to stock quotes they are interested in, and then receive market information for the subscribed companies in real time.	A person subscribes to information of interest, and then receives such information in real time.
consolidated database	Every time a purchase is made at Target, the point-of-sale system records the data locally, and a separate program is triggered to forward the information on to a central database for all of Target's stores.	Every time a database transaction is written to the database, a separate process is triggered which copies the transaction to a larger database which consolidates information from several sources.
store & forward	George handed the package to Sally, and told her to give it to Jim whenever she saw him next. She saw Jim later that day and handed it to him.	Information is transmitted from A to B, and then onto C when B comes within range of C.
lock & unlock	When I talk to Continental airlines, the call agent locks the database while I make my decision about which seat to take on the airplane: as soon as I decide she reserves the seat and releases the lock.	Whenever a request is made to update a certain type of record, the database table is locked, and only unlocked upon the completion of the update.

FIGURES

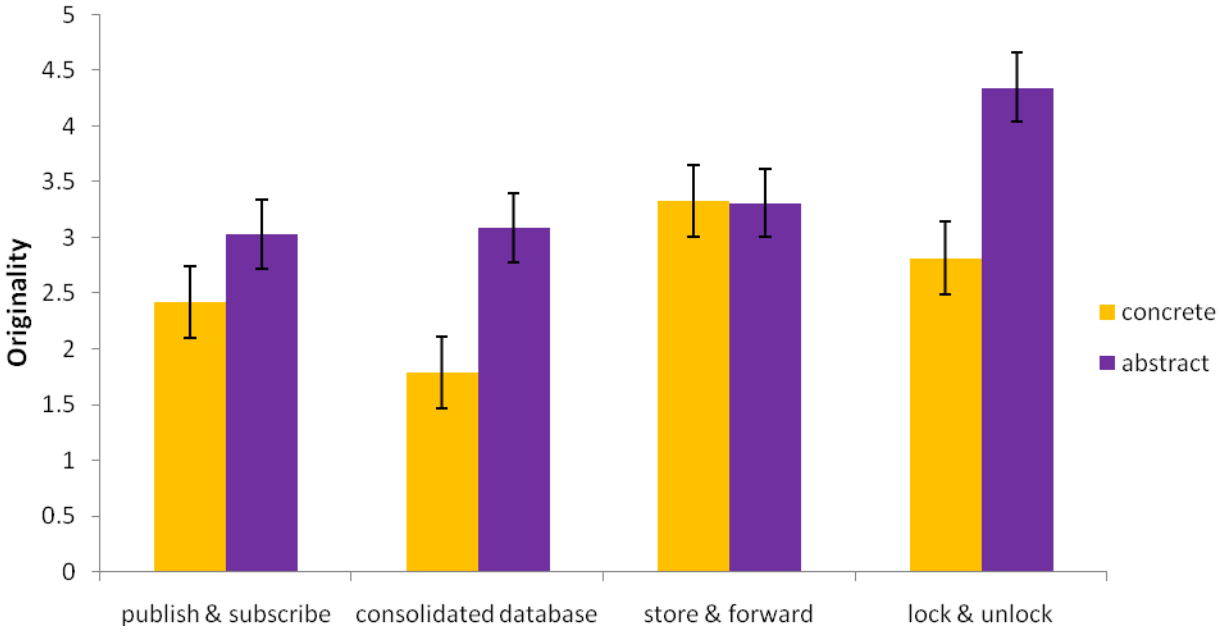

Figure 1: Mean originality score by abstractness

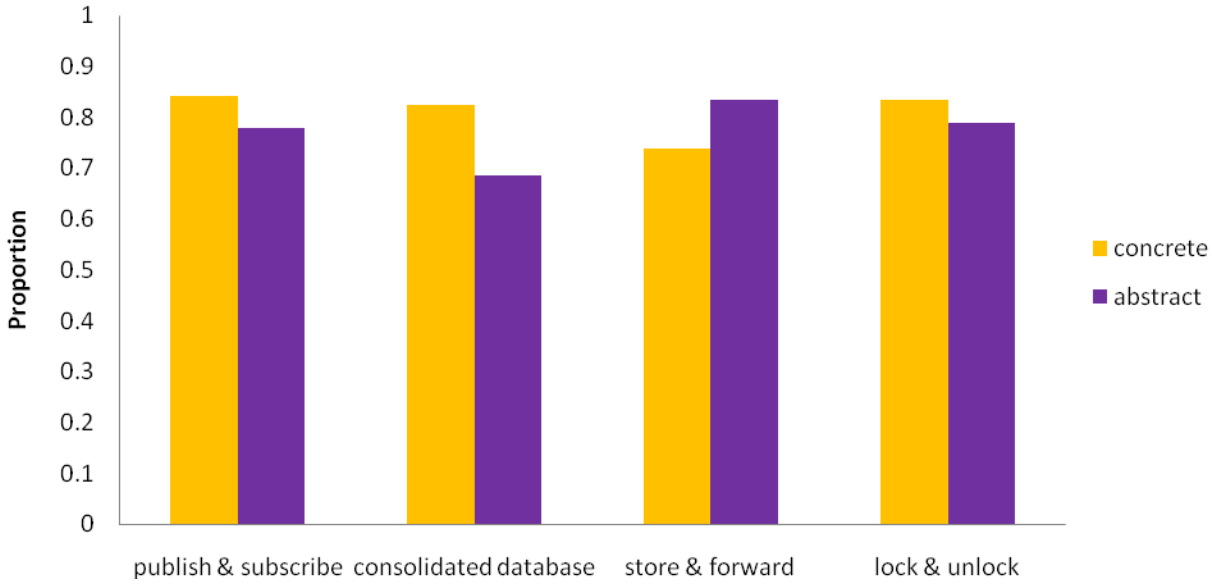

Figure 2: Proportion of solutions that fit problem constraints by abstractness

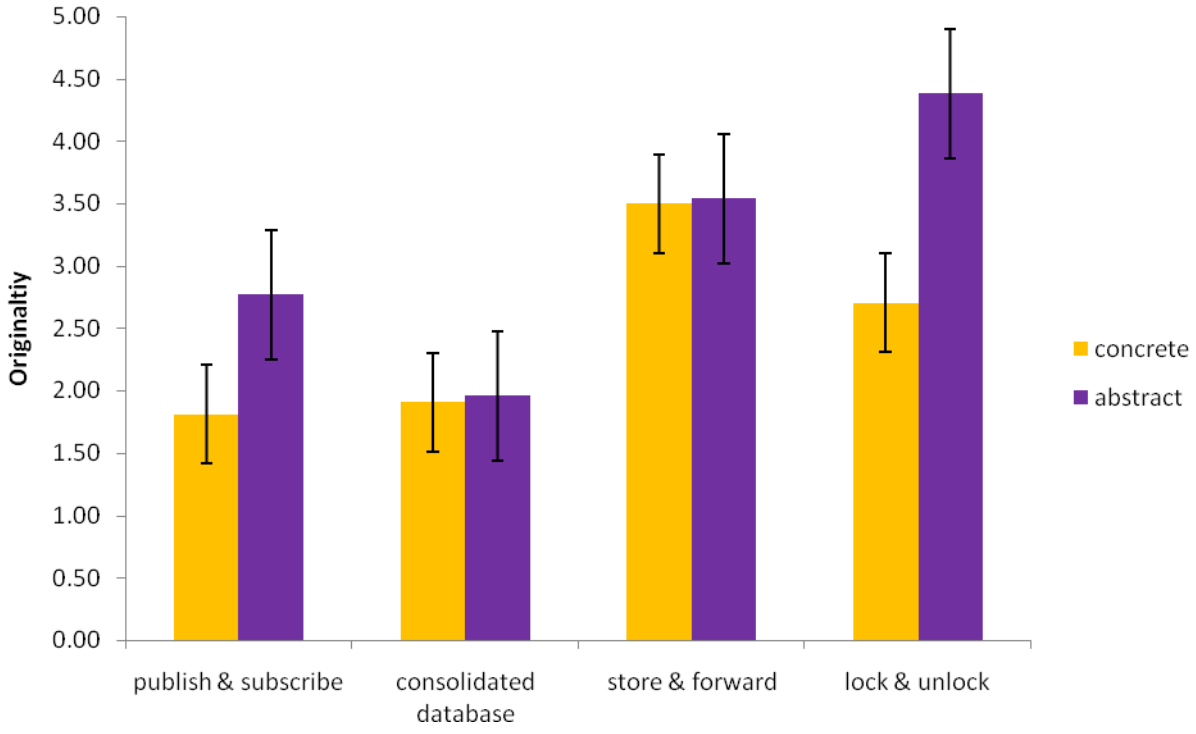

Figure3: Mean originality score for solutions that fit the problem constraints by abstractness

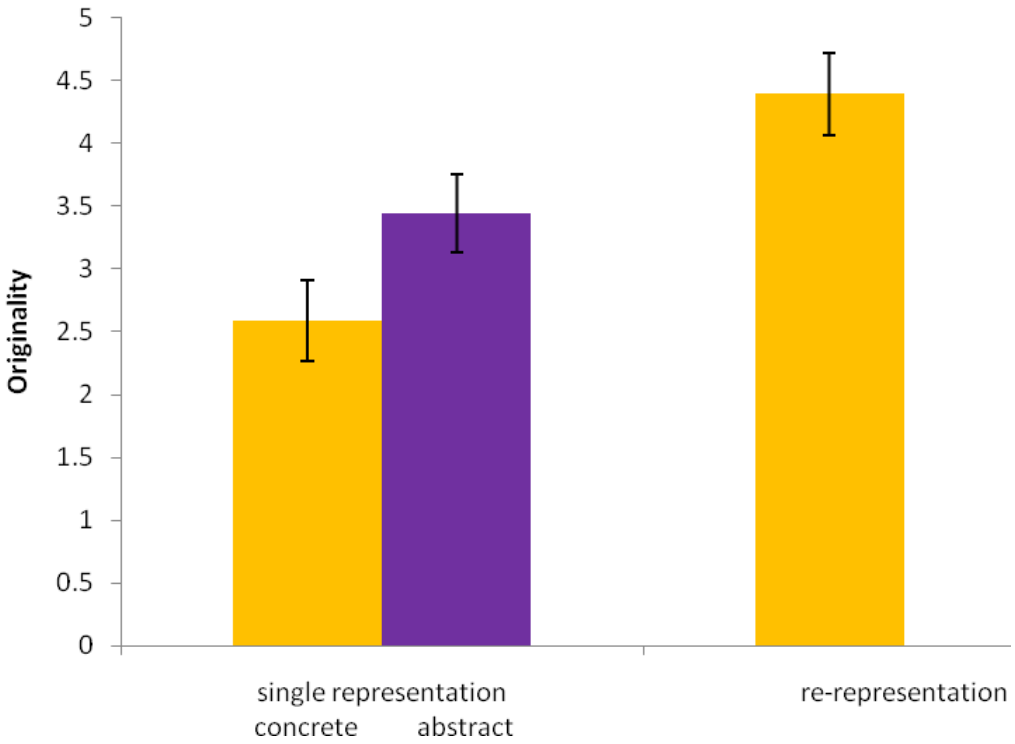

Figure 4. Mean originality score. The two left bars represent the mean originality scores for the concrete and abstract versions of the first four problems. The bar on the right is the mean originality score for the solutions generated for "tracking" problem.

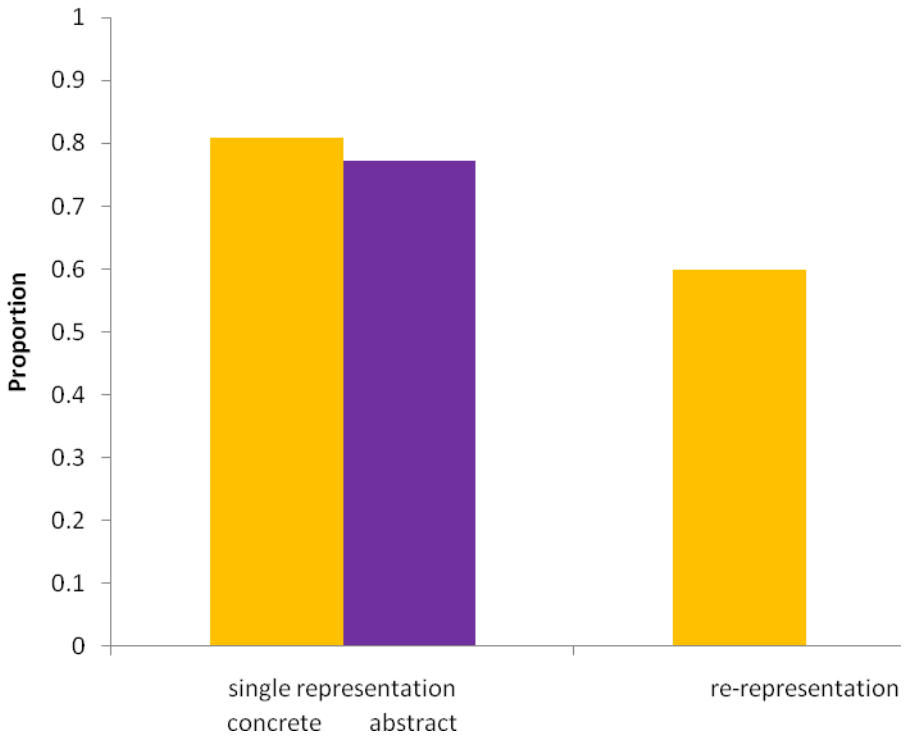

Figure 5: Proportion of solutions that fit problem constraints. The two left bars represent proportion of correct solutions for the concrete and abstract versions of the first four problems. The bar on the right is the proportion of correct solutions for the solutions generated for "tracking" problem.

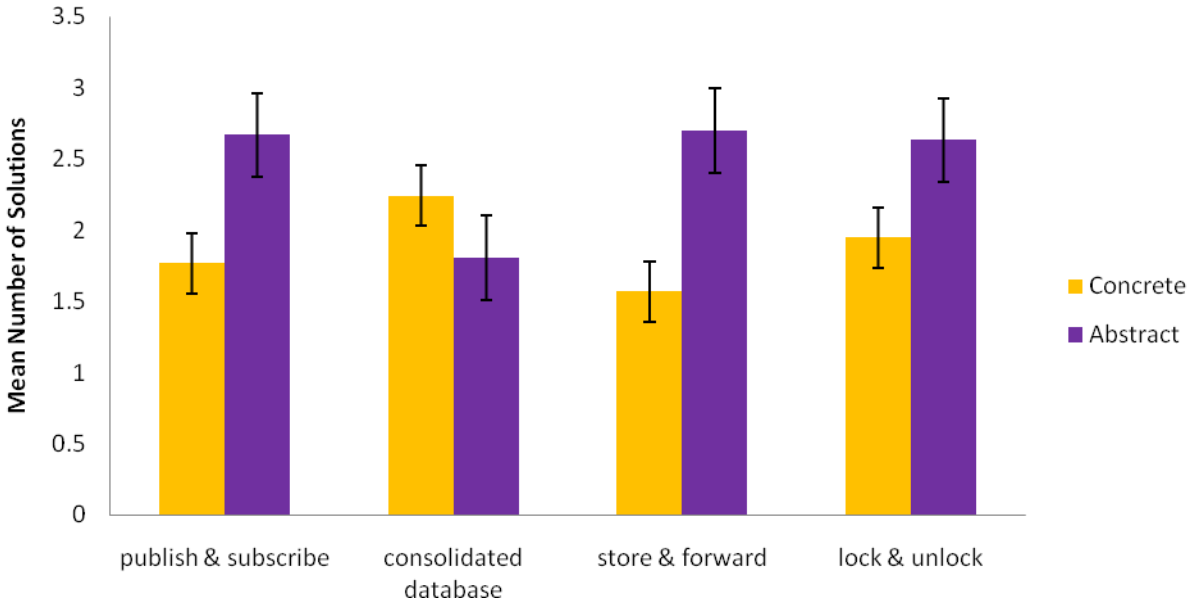

Figure 6: Mean number of participant-generated solutions by abstractness.

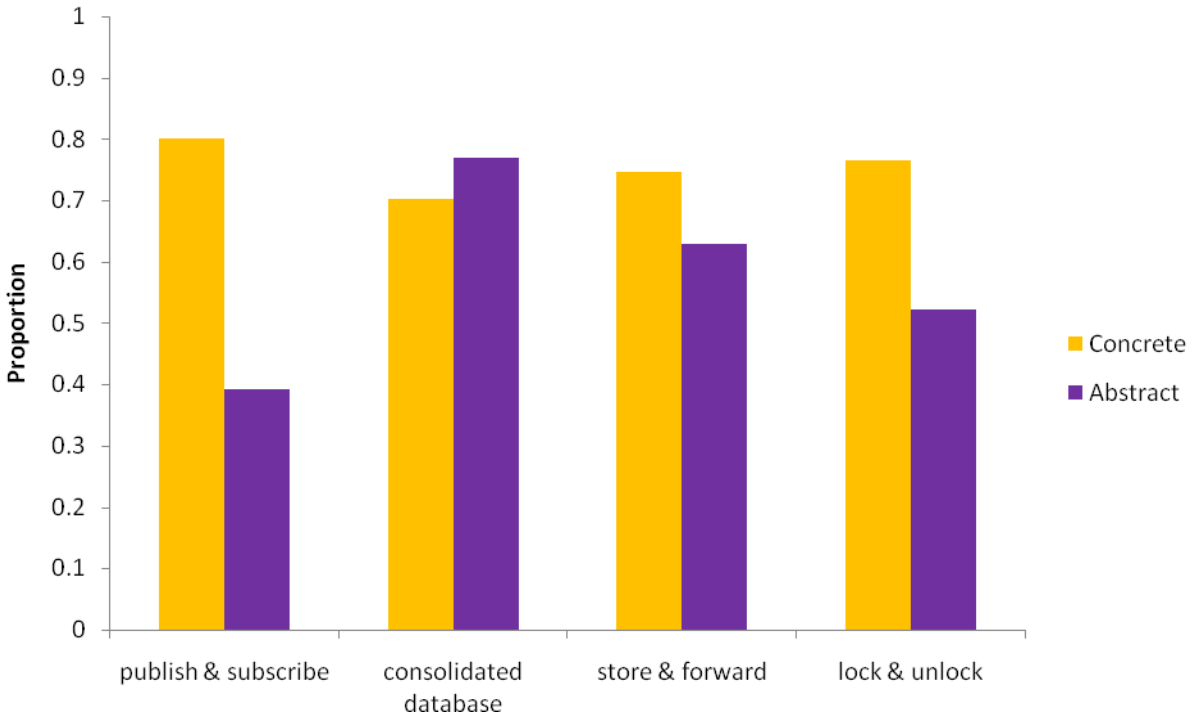

Figure 7: Proportion of solutions that fit problem constraints by abstractness

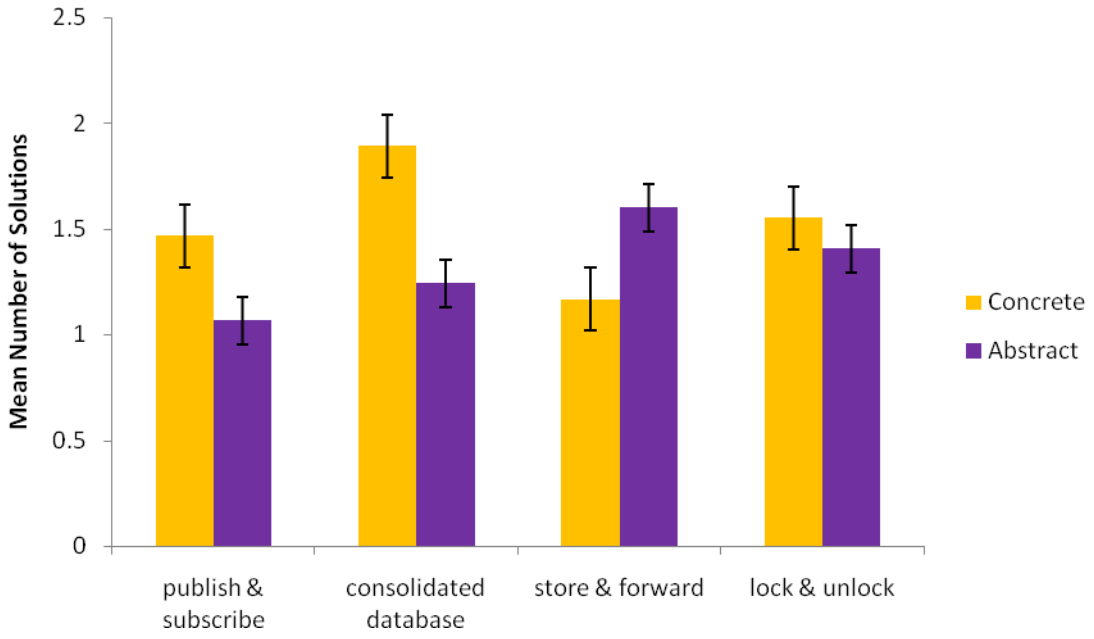

Figure 8: Mean number of participant-generated solutions that fit the problem constraint by abstractness.