Artificial Intelligence for Engineering Design, Analysis and Manufacturing (2010), *24*, 231–244.
© Cambridge University Press, 2010 0890-0604/10 $25.00
doi:10.1017/S0890060410000077

A fix for fixation? Rerepresenting and abstracting as creative processes in the design of information systems

DORIS ZAHNER,[1] JEFFREY V. NICKERSON,[1] BARBARA TVERSKY,[2] JAMES E. CORTER,[2] AND JING MA[1]

[1]Stevens Institute of Technology, Castle Point on Hudson, Hoboken, New Jersey, USA
[2]Teachers College, Columbia University, New York, New York, USA

(Received April 15, 2009; Accepted October 15, 2009)

Abstract

Fixation prevents the associations that are bridges to new designs. The inability to see alternative solutions, or even to see how to map known solutions onto current problems, is a particularly acute problem in the design of software-intensive systems. Here, we explored two related ways of liberating fixated thinking: abstracting and rerepresenting. Although both techniques helped designers generate original ideas, not all the added ideas fit the problem constraints. We discuss ways the results might be used to generate reflective design aids that help designers to first generate original ideas and later prune them.

Keywords: Abstraction; Design of Information Systems; Fluency: Originality; Rerepresentation

1. INTRODUCTION

How does thinking happen? At the core, thinking is associationistic (e.g., Rumelhart & McClelland, 1986; Anderson, 1993), that is, 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. 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. 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 how the mind works naturally, both in memory (Rundus & Atkinson, 1970) and in judgment (Sloman, 1996; Kahneman & Frederick, 2005): first a burst of rapid associations, then a slow evaluation of them. New designs are often created by adapting previous ones, a process called *transfer*. In one set of studies, students read a problem about radiation, where the goal is to destroy a tumor in the body without destroying surrounding tissue (Dunker, 1945). They also read the solution, which is to send weak rays from many directions that converge on the tumor. After performing irrelevant tasks, students were presented with analogous problems in other domains, capturing a castle or putting out a fire, where success depended on converging from many directions. Students typically failed to transfer a solution from one domain to another, especially when those domains were far or remote from the solution domain (Gick & Holyoak, 1980, 1983). Within engineering design, some research suggests that transfer, both near and far, can happen during the design process (Christensen & Schunn, 2004). However, most studies lead to a more pessimistic conclusion. Unfortunately, research in general problem solving, a superset of design problem solving (Simon, 1995), has repeatedly demonstrated failures to transfer known solutions to new problems (Gick & Holyoak, 1980, 1983; Ross, 1989; Novick, 1990; Ross & Kennedy, 1990; Novick & Hmelo, 1994). The typical reason for the failure to apply old solutions to solve new problems is a failure to see the similarity of the global constraints or the deep structure of the problems. The reason for this failure is, in turn, a general property of associationistic thought: it travels most easily along

Reprint requests to: Doris Zahner, Teachers College, Columbia University, 525 West 120th Street, Box 118, New York, NY 10027. E-mail: dwc14@columbia.edu

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 in which a fire in a field of oil wells needs to be put out, but the field of oil wells is too dense to allow enough retardant to be delivered along any one path. Tumors in the body and oil wells in Saudi Arabia are very different 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, each new alternative did not stray far from the previous one (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 (Schön, 1983; Goldschmidt, 1991, 1994; 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 rerepresent 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. However, efficiently training this facility has been elusive. One requirement is abstraction, both to the problem at hand and from previously encountered problems. In some learning tasks, 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 (cf. Southey, 1837). When problems are too concrete and specific, so is transfer. Yet, when problems are presented too abstractly, without any domain instantiation, successful solutions decrease (Wason & Johnson-Laird, 1972; Holyoak & Cheng, 1995). Thus, 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. 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.

To measure the creativity of designers' solutions, we adapted previously proposed models of creativity (Finke, 1990; Maher, 2008) where creativity is measured by score on two scales. The first scale measures originality or novelty, and the second scale evaluates practicality, usefulness, or correctness.

Here, designers were asked to take prototypical information system configurations (e.g. Gamma et al., 1995) and find other situations for which those configurations are appropriate. Thus, they needed 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 is unoriginal to merely change FedEx to UPS while solving a problem involving shipping. Syntactic substitutions might be correct but will be 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, a form of practicality. 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 (Wallach & Kogan, 1965; Amabile, 1996; Gasper, 2004).

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, rerepresenting it in a more abstract fashion. This term, rerepresentation, 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 (Oxman, 1997) and artifact design (Visser, 1991), in which multiple and varied representations are important. 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 rerepresented at a higher level of abstraction, it becomes easier for the designer 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 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. In contrast, expert designers do rerepresent their own work: in architecture, they sketch, and then observe the sketch anew, finding new interpretations (e.g., Schön, 1983; Goldschmidt, 1991; Suwa & Tversky, 1997; Suwa et al., 2001). It is plausible that expert designers perform a series of sketches to abstract from the original situation, thereby encouraging freer association. Thus, we predict that the process of rerepresenting a concrete situation will also contribute to generating original solutions. It is even possible that the rerepresentation process itself is more important than the rerepresentational medium. That is, textual rerepresentation may work about as well as diagrammatic rerepresentation, although at some level, diagrammatic and linguistic representations of ideas are bound to inspire different thoughts and inferences.

2. ORIGINALITY AND FIT EXPERIMENT

In this study, participants were presented with a set of text scenarios describing potential design problems, such as a database problem where the database needs to be locked for a set period of time (see Table 1). The text descriptions varied in abstractness. For each of four problems, participants were given the following instructions: "For this solution scenario, create another scenario that is structurally similar, but applies to a very different situation." 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 solutions from one domain to another (cf. Hamming, 1986; Bergman et al., 2002).

2.1. Method

2.1.1. Participants

Thirty-nine students (10 women, 29 men) 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. Sixty-six percent of them were full-time students. The full-time students had an average of 0.75 years (SD = 1.08) of full-time working experience, whereas the part-time students had an average of 5.6 years (SD = 3.27) of full-time working experience.

2.1.2. 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, and form B the reverse. The text for the fifth problem had only a single version. Instead of varying abstractness, we varied the medium of rerepresentation: participants were instructed to first rerepresent the problem in an abstract form (either textually or diagrammatically, according to condition), then to create a structurally similar but different text scenario

Table 1. *The four different problem topics with their concrete or abstract variants*

Topic	Concrete	Abstract
Publish and 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 Target stores.	Every time a database transaction is written to the database, a separate process is triggered that copies the transaction to a larger database that consolidates information from several sources.
Store and 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 and 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.

2.1.3. Coding

Two raters rated each solution on two components of creativity: first, originality, and second, whether the solution fit the constraints of the given problem. One rater was a doctoral student with a background in information systems. The other rater was a postdoctoral researcher with a background in psychology. Originality was rated on a 1–9 scale, where 1 indicates *extremely unoriginal* and 9 indicates *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 averaged ratings. A total of 10 matrices were created, 1 for each version (abstract or concrete) for the first four problems, and 2 matrices for the fifth problem, created by dividing participants' solutions according to whether they created a text- or diagram-based rerepresentation as an intermediate step. The interrater reliability was calculated for each of the matrices; for all problems, $\alpha > 0.95$.

As a manipulation check, four judges rated the dissimilarity between the two presented versions of each problem. Two of the four judges were psychology doctoral students, one was a professor of psychology, and one was a professor of information systems. Dissimilarity was also coded on a scale from 1 to 9, where 1 is *extremely similar* and 9 is *extremely dissimilar*. The average rating was used for all analyses.

The second aspect of each solution that was coded was whether the structure of the participant solutions matched the structure of the canonical problem text. If a participant's solution corresponded to the same network topology as described by the canonical text, it was rated as a match and given a score of 1. If a solution did not correspond to 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 problems are analogous. If the solution is not a structural match to the given problem, the participants either do not completely understand or they fail to map the problem constraints, so that their solutions, although appearing to be novel, are insufficient. Two raters rated the solutions for matching the problem constraints, with an interrater reliability score of $\alpha = 0.94$. 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 classify the problem solutions (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 & 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. Because of a disaster, Julie transmits an e-mail about her health to her family. The e-mail does not get there because of failures of communication infrastructure. However, the e-mail gets collected by a nearby car, which, when it travels near the house of Julie, transmits the e-mail to her home/family.
3. I watch the news on TV every day. My friend does not 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.

2.2. Results

2.2.1. 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 nonredundant ideas) and flexibility (number of domains represented in the ideas). We expected that participants in the more abstract conditions would have greater levels of fluency and flexibility, that is, more nonredundant, unique domains. On average, for all four problems, the number of unique domains at both the general and the specific levels was greater for the abstract conditions than for the concrete conditions.

2.2.2. Originality of solutions

We also expected that the abstract versions of the problem would produce more original solutions than the concrete versions. To test this, we calculated the mean (and standard deviation) dissimilarity scores 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 & forward* problem, the originality scores for the abstract condition solutions were higher than those for the concrete condition. A split-plot factorial analysis of variance was run on the data and results show that the solutions created under the abstract condition (mean = 3.45, SD = 0.62) were significantly more original than those created under the concrete condition (mean = 2.59, SD = 0.65) across all four problems [$F(1, 34) = 4.23$; $p < 0.05$].

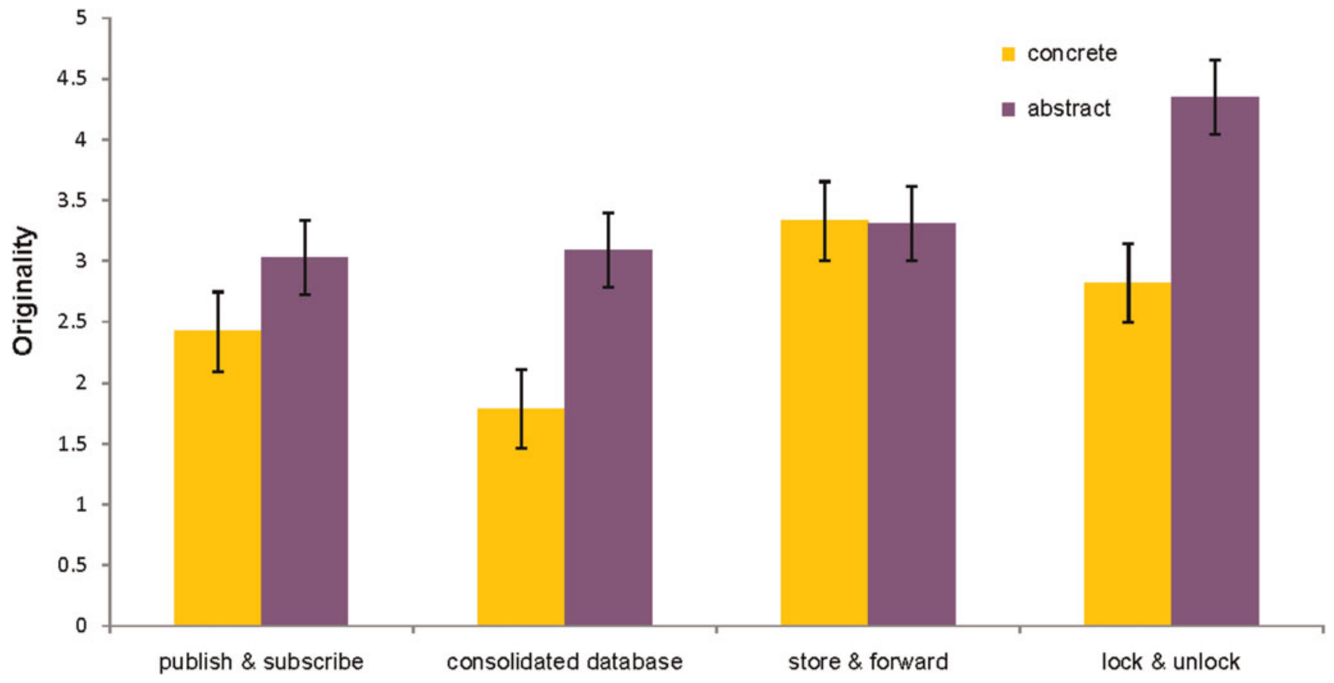

Fig. 1. The mean originality score by abstractness. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

The *store & 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 & forward* problem was qualitatively different from the other three. 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 & forward* problem to have the largest dissimilarity between the two presented problem versions. The abstract and concrete versions of the *publish & subscribe* problems were rated as having a dissimilarity of 3.5. The two versions of the *consolidated database* problems were rated as having a dissimilarity of 3.75. The two versions of the *database lock & unlock* problems were rated as having a dissimilarity of 4.25. In contrast, the two versions of the *store & forward* problems were given a dissimilarity rating a point higher (5.25) than the next most dissimilar

Upon further inspection of the problem texts, we noticed that the *store & forward* problem was the only one that had only *people* in the concrete condition as opposed to machines, suggesting a possible resolution. People are perceived to vary on many more features than machines, thus enabling a greater variety of solutions from social scenarios than from machine scenarios, in the end yielding a set of solutions that appear more original. Of the 19 solutions generated from the concrete condition of this problem, it was interesting that 15 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 because 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.

2.2.3. *Fitting the problem constraints*

For this study, solutions were evaluated in two ways, originality and fit. Because the thought processes that lead to original solutions are different from the thought processes that ensure fit, originality and fit are not necessarily correlated. There are cases of highly original solutions that do not fit the problem constraints and cases of solutions that fit the problem constraints but 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 is evident in the graph, the concrete condition yielded more solutions that fit the problem constraints than the abstract condition, with the exception of the *store & forward* problem. This result is the exact complement of the results from the analyses of origi-

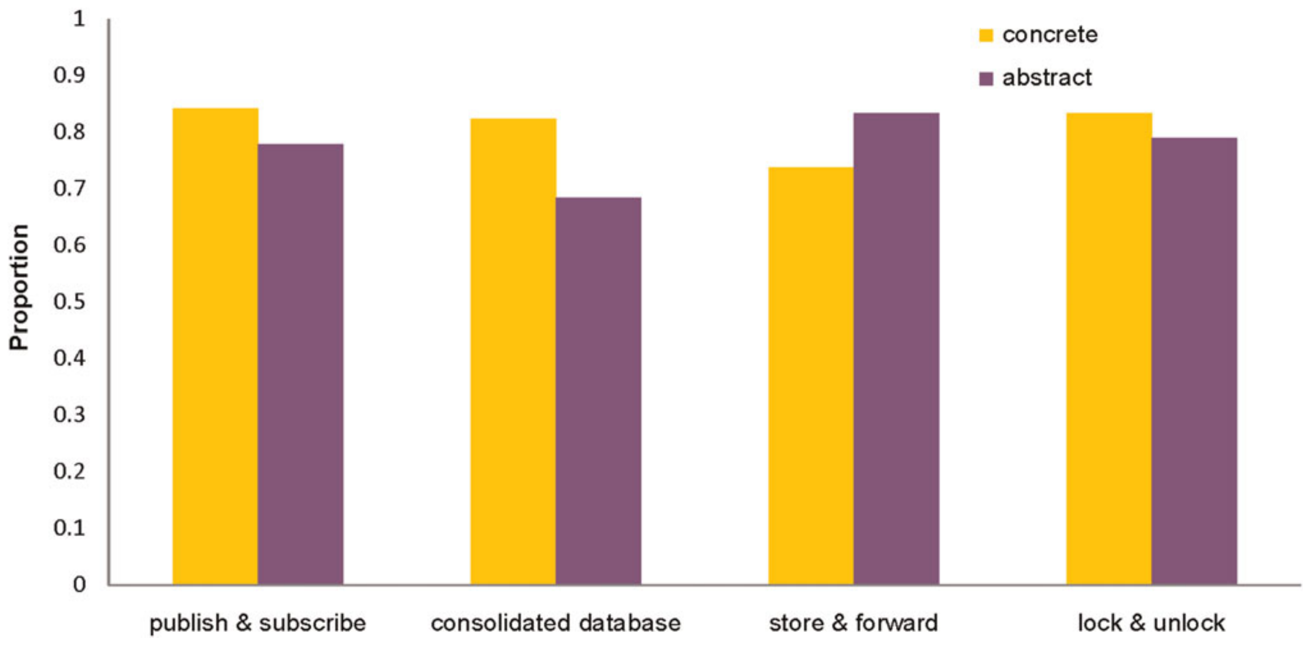

Fig. 2. The proportion of solutions that fit problem constraints by abstractness. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

nality (Fig. 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. The correlation between originality of the solution and fitting the problem constraint is $r(147) = -0.44$ ($p < 0.001$), meaning that the more original the solution is, the less likely it is to be correct.

In addition to these analyses conducted on all solutions, we investigated the originality score for only those solutions that fit the problem constraints. 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 minimal syntactic substitutions, whereas the correct solutions in the abstract condition are both highly original and fit the problem constraints.

2.2.4. *Rerepresenting: 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 reading the 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. The diagram condition asked participants to create an abstract structural diagram (cf. Fowler, 2004) of the given problem text. 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 medium of the generated solution, text or diagram, had no effect on the originality of the final scenario that is created. However, it was expected that rerepresenting the canonical text would induce participants to generate solutions that were at least as original as the abstract conditions and more original than the solutions from the concrete condition. Indeed, as shown in Figure 4, we found that the mean originality score for the rerepresented problem (mean = 4.45, SD = 1.72) was significantly higher than both the abstract solutions [mean = 3.45, SD = 0.62, t (105) = 3.96, $p < 0.001$] and the concrete solutions [mean = 2.59, SD = 0.65, t (104) = 8.68, $p < 0.001$]. Therefore, rerepresenting works at least as well as presenting an abstraction at the start. It works significantly better for generating more original solutions.

This result suggests that asking students to abstract from concrete examples first, and then generate analogs, is a good way to encourage originality. It does not seem to matter whether the abstraction is expressed in words or as a diagram. Asking people to rerepresent 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 and as before, the original solutions are often incorrect, indicated by a negative correlation between originality and fit [$r(32) = -0.61, p < 0.001$]. The proportion of correct solutions from the *tracking* problem is lower than from the other four problems. One intriguing find-

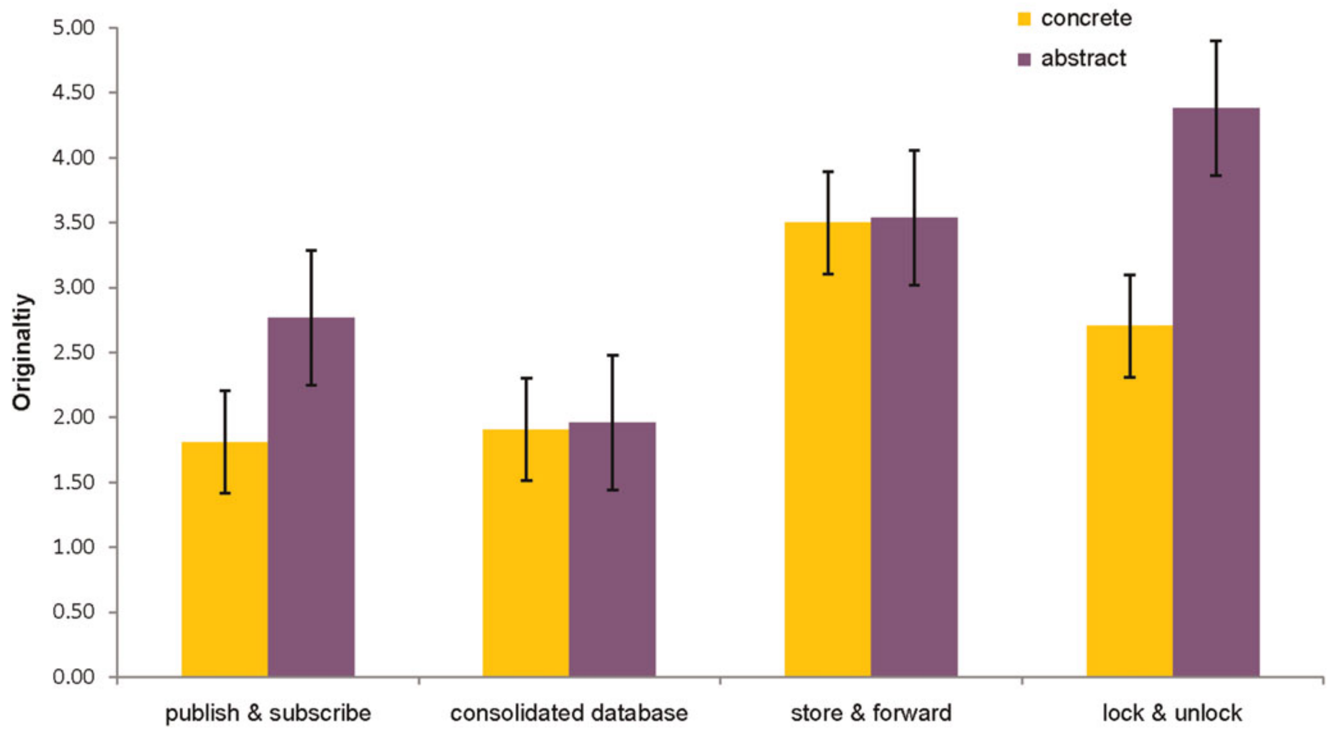

Fig. 3. The mean originality score for solutions that fit the problem constraints by abstractness. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

ing is that the diagram condition yielded slightly more correct solutions than the text condition.

2.2.5. Qualitative analysis

In addition to the quantitative analyses reported above regarding originality and fit, we explored 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. As before, 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 so-

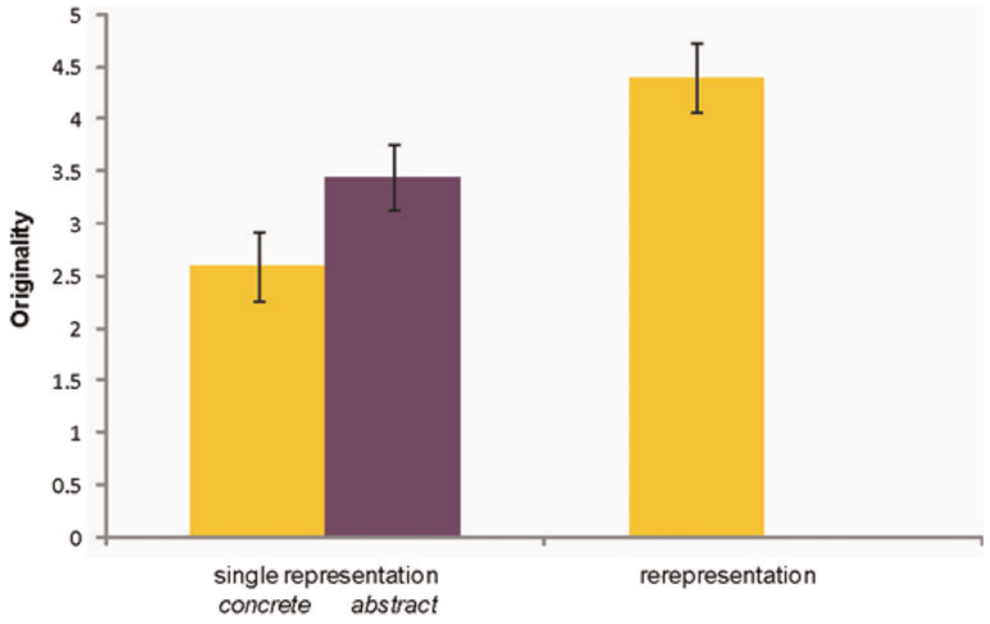

Fig. 4. The mean originality score. The two bars on the left represent the mean originality scores for the concrete and abstract versions of the first four problems. The bar on the right is the mean originality score for the solutions generated for the tracking problem. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

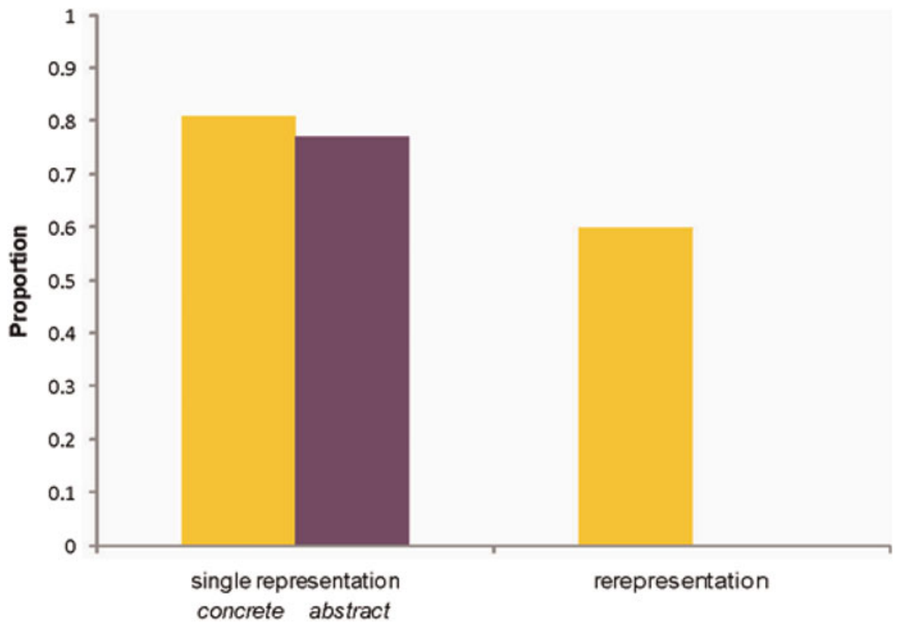

Fig. 5. The proportion of solutions that fit problem constraints. The two bars on the left represent the proportion of correct solutions for the concrete and abstract versions of the first four problems. The bar on the right is the proportion of correct solutions for the solutions generated for the tracking problem. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

lutions 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 & subscribe* problem is the following:

> 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 [the] 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 received high scores for originality because they were situated in remote domains. However, none of the solutions fit the original problem constraints of *publish & 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 constraints. For example, for the *lock & 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[ly done the] 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 while matching the problem constraints.

An informal class discussion following the experiment yielded interesting insights 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 in 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.

3. FLUENCY EXPERIMENT

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

In this experiment we asked participants to brainstorm and come up with 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.

3.1. Method

3.1.1. 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 30 subjects for each question. Two hundred fifty-six subjects participated in this study (101 females, 155 males), and they were compensated with a nominal stipend. Their ages ranged from 17 to 69, with an average age of 31 ($s = 10.92$ years). They spent an average of 2 min 12 s completing the task and a related demographic survey ($s =$ 2 min 35 s). 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.

3.1.2. Materials

The two versions of the four problems from the first study (Table 1) were used for this second study, with one change. Because the *store & forward* problem yielded anomalous results in the first study, perhaps because the concrete version involved no machines, we changed the concrete version of the *store & 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."

3.1.3. 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. We also coded the solutions, as we did in the first experiment, for whether they matched the problem constraints. Because the interrater reliability for fit was over 0.90 for the first experiment, we only used one of the two original coders to rate the solution fit for this experiment.

3.2. Results

3.2.1. Fluency

We predicted that the participants would create more alternative solutions for the abstract versions than for the concrete versions of the problems. 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 (mean = 2.45, SD = 0.43) created significantly more solutions than those in the concrete condition (mean = 1.88, SD = 0.29) across all four problems [F (1, 254) = 9.49, $p < 0.01$].

As a result of changing the *store & 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 pre-

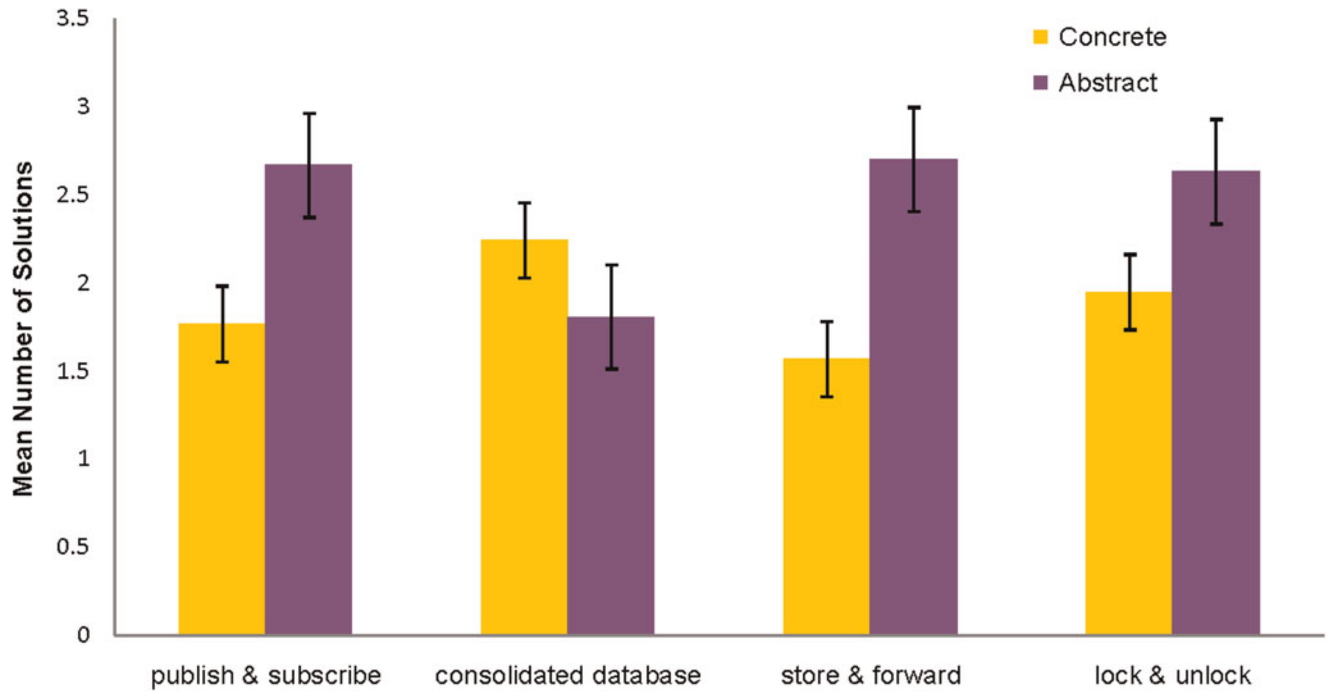

Fig. 6. The mean number of participant-generated solutions by abstractness. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

vious 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 even 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.

3.2.2. *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.

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 & 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.

4. 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 the classroom experiment, 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. The abstract versions were not devoid of content, but were not specific to a domain, whereas the concrete versions were domain specific. In the public experiment, participants from varying backgrounds generated as many similarly structured scenarios as possible from the problem they were given, stated relatively abstractly or relatively concretely. The question of interest was whether the abstractness of the presented cases would influence the originality and fluency of proposed solutions, as well as the fit to problem constraints. Both experiments found that abstractness promoted original ideas in the design of information systems. In the classroom study, solutions generated from the abstract version were significantly less similar in domain and content from those generated from the concrete condition, consistent with results from a previous study (Tversky et al., 2008). In the public experiment, participants generated more solutions

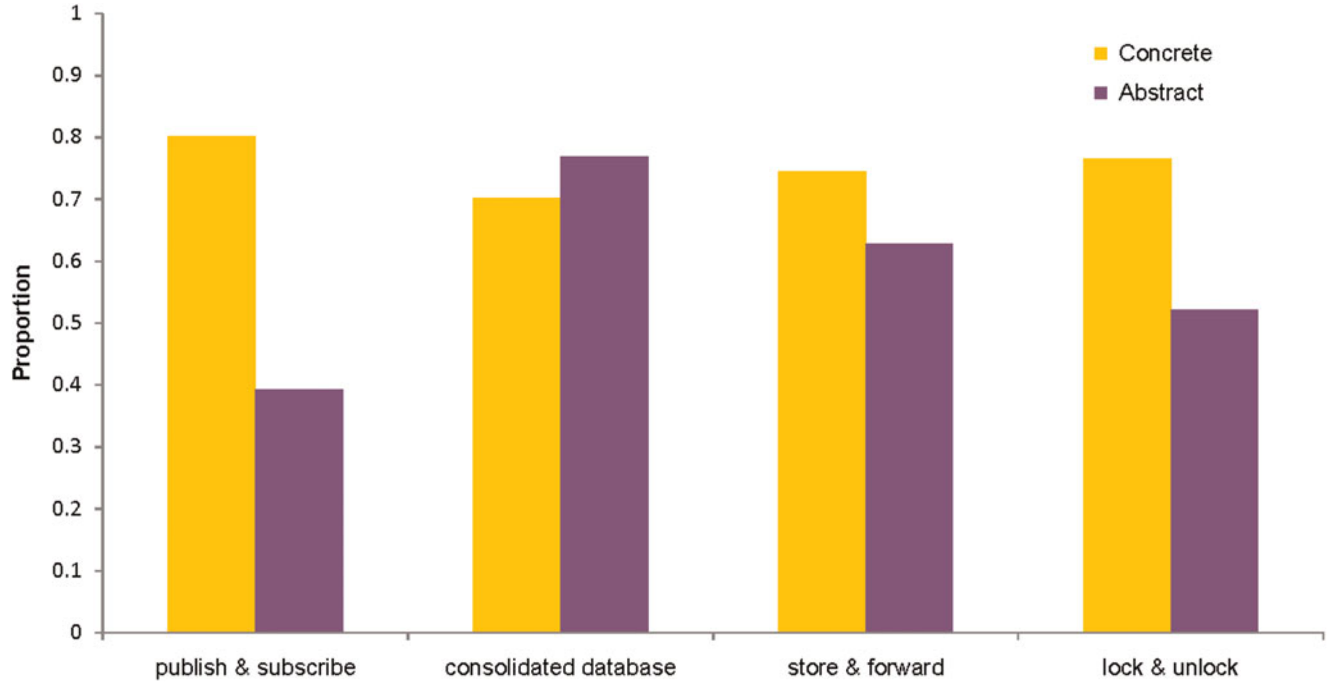

Fig. 7. The proportion of solutions that fit problem constraints by abstractness. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

to the abstract version than to the concrete version. Creativity was assessed by several measures, and in particular, the match to problem constraints was assessed.

Solutions that were considered highly original tended to come from the abstract versions of the problems, whereas solutions that were considered to be unoriginal tended to come from the concrete versions. Participants provided with the abstract versions generated more solutions as well as solutions from a broader range of domains (Fig. 6). Thus, abstraction augments divergent thinking. Remember that the abstract examples used

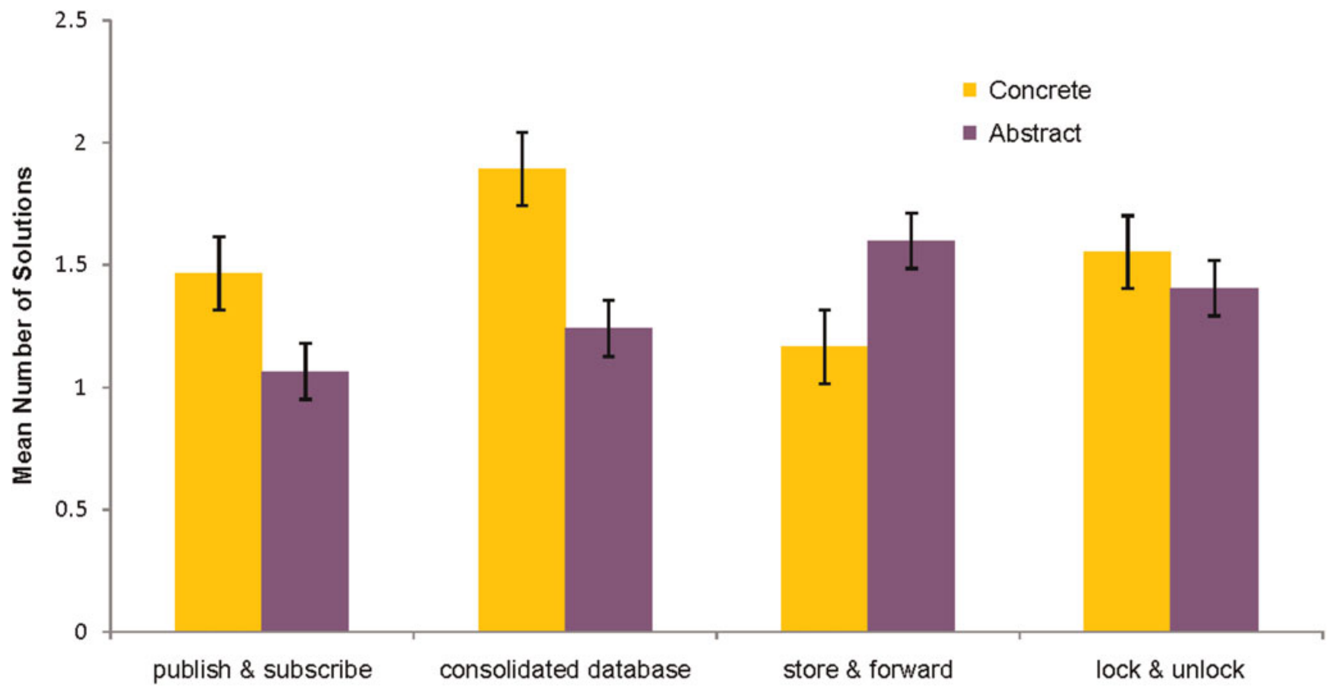

Fig. 8. The mean number of participant-generated solutions that fit the problem constraint by abstractness. [A color version of this figure can be viewed online at journals.cambridge.org/aie]

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 solutions does have a cost. A smaller proportion of them actually fit the problem constraints so that the actual number of solutions fulfilling the problem constraints was not greater for the abstract version than for the concrete version (Figs. 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 immediately useful. However, some may be almost right so that they can be revised to comply with the constraints. Evaluation as a second stage can eliminate or tweak the solutions that do not fit. One challenge is to find ways to formulate abstract problems to make the constraints salient and 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 classroom experiment, participants were encouraged to create an abstract solution by themselves, rather than being provided one, a process termed rerepresentation. Rerepresentation 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 generate an abstract solution and then generate concrete solutions. When participants were given a concrete version and asked to create an interim abstract version, either in text or diagram form, the ultimate concrete solutions were more original than those of abstract and concrete conditions from the other problems. This finding echoes the previous findings in transfer of known problem solutions to other domains. For that research, transfer to other domains was facilitated by both multiple examples and by generating an abstract rule, expressed either as a diagram or a sentence (Gick & Holyoak, 1980, 1983). The act of generating an abstract case seems to encourage developing a deeper understanding, or a mental model, of the situation (e.g., Gentner & Stevens, 1983; Ross, 1989; Kotovsky & Gentner, 1996; Gentner & Wolff, 2000). Once a person has an adequate mental model, it is easier to see structural similarities (Gentner & Markman, 2006) among different situations. This structure mapping promotes transfer, and, in the field of design, may lead to more solutions.

The process of rerepresentation 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 a stage that includes evaluating, tweaking, 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 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 rerepresentation 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 based on detailed, concrete 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 findings suggest that adding a design step in which requirements are abstracted may be useful, as it may encourage associations that map new problems 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 (Schwartz & Black, 1996; Goldstone & Son, 2005). Fading can be accomplished by selectively removing labels from sequence diagrams until the domain associations have been reduced. In this way, the designer gradually loosens the constraints and expands the possibility of analogical transfer. The convergent thinking component (the checking of such 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 a human-generated analog, and propose a mapping to the users. If the system correctly finds a practical mapping, this will assist the designers by confirming their intuitions. If the system finds only impractical alternatives, it may cause the designers to reevaluate analogies, the same way scientists reevaluate theories (cf. Lee & Nickerson, 2010). Either way, the outcome should stimulate appropriately focused design thinking.

5. IMPLICATIONS

Software-intensive systems are complex artifacts, which are notoriously difficult to construct. Little is known about how to design them (cf. Lee, 2000). To find solutions, designers think by following associations in order to generate possible solutions and by attempting to map known solutions to new problems (cf. Bergman et al., 2002). However, fixation is a common block in many creative processes. The present research has shown that more abstract formulations of problems can free designers from fixation, leading to more original solutions. However, designers are often charged with designing specific, concrete examples. In this study we tested a technique that follows standard practice and initially provides designers with concrete examples. Then, parting from tradition, designers were asked to generate an abstraction, either textual or diagrammatic, before generating analogs to the original examples, a process of rerepresentation. This procedure yielded a greater diversity of solutions than from the typical concrete scenarios. It is important that rerepresenting generated a greater diversity of solutions than an initial abstract scenario. Rerepresenting appears to be a powerful technique for generating novel solutions. It is also a practical technique, because it can be used by any designer for any design task.

However, novel solutions do not necessarily conform to the constraints of the problem. Surprisingly, these experiments consistently yielded negative correlations between originality and fit. The kinds of thought processes that yield remote and creative solutions appear to be antithetical to the kinds of thought processes that yield solutions that conform to constraints. Abstraction and rerepresentation are thus important ways of generating novelty, but they also generate error. As for other techniques of divergent thinking, abstraction and rerepresentation need to be followed by a later editing to remove or modify insufficient solutions. One unanswered question here is whether the "incorrect" divergent solutions were almost correct, that is, whether they could be sensibly revised to conform to constraints. If so, methods to encourage divergent thinking, like abstraction and rerepresentation, are all the more valuable.

These techniques that encourage original solutions 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. This would encourage designers to rerepresent their ideas through successive abstractions. In addition, a tool might support the verification of generated ideas by explicitly searching for and representing the detailed mapping between problems in one domain and solutions in another.

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

ACKNOWLEDGMENTS

We are grateful to Yin Jin Rho and Mike McGahan for assistance on various aspects of the project. This work was supported by National Science Foundation Grants NSF IIS-0855995, NSF IIS-0725223, and 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: Erlbaum.

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.

Christensen, B.T., & Schunn, C.D. (2004). The relationship of analogical distance to analogical function and preinventive structure: the case of engineering design. *Memory & Cognition 35(1)*, 29–38.

Dunker, K. (1945). On problem solving. *Psychological Monographs 55(No. 270).*

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

Fowler, M. (2004). *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Reading, MA: Addison–Wesley.

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. *Journal of Cognitive Science 6(1)*, 1–20.

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

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. Hillsdale, NJ: Erlbaum.

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 Studies 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. *Journal of the Learning Sciences 14(1)*, 69–110.

Hamming, R.W. (1986). You and your research. *Proc. 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., & Siefert, 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.

Kahneman, D., & Frederick, S. (2005). A model of heuristic judgment. In *Cambridge Handbook of Thinking and Reasoning* (Holyoak, K., & Morrison, R., Eds.). Cambridge: Cambridge 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.S. (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. *Proc. 11th Int. Conf. Information Management.* Accessed at http://www.people.vcu.edu/∼aslee/ICIM-keynote-2000

Lee, A.S., & Nickerson, J.V. (2010). Theory as a case of design: lessons for design from the philosophy of science. *Proc. 43rd Annual Hawaii Int. Conf. Systems Science.*

Maher, M.L. (2008). CreativeIT: does IT enhance creativity or does creativity enhance IT? Keynote Address. *Proc. Design, Computing, and Cognition Conf.*

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. *Proc. 29th Int. Conf. Information Systems.*

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*, 3rd ed. New York: Charles Scribner's Sons.

Oxman, R. (1997). Design by re-representation: a model of visual reasoning in design. *Design Science 18(4)*, 329–347.

Redmiles, D., & Nakakoji, K. (2004). Supporting reflective practitioners. *Proc. 9th Int. Conf. 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, and Cognition 16*, 42–55.

Rumelhart, D.E., & McClelland, J.L. (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1. 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.

Schwartz, D.L., & Black, J.B. (1996). Shuttling between depictive models and abstract rules: induction and fallback. *Cognitive Science 20(4)*, 457–497.

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.

Southey, R. (1837). The story of the three bears. In *The Doctor*, Vol. 4. London: Longman.

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 *Proc. 25th Annual Conf. 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.

Visser, W. (1991). Evocation and elaboration of solutions: different types of problem-solving actions. An empirical study on the design of an aerospace artifact. In *Cognitiva 90. At the Crossroads of Artificial Intelligence, Cognitive Science and Neuroscience. Proc. 3rd COGNITIVA Symp.* (Kohonen, T., & Fogelman-Soulié, F., Eds.). Amsterdam: Elsevier.

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.

Doris Zahner is an Adjunct Assistant Professor of psychology and education at Teachers College, Columbia University. She received her PhD in cognitive psychology in 2005 from Teachers College, Columbia University, and her BS in educational psychology in 1998 from Cornell University. Dr. Zahner's research involves the use of diagrams in problem solving, particularly in the domain of probability.

Jeffrey V. Nickerson is an Associate Professor and Director of the Center for Decision Technologies at Stevens Institute of Technology. Dr. Nickerson's research involves the design of networks of all types. His research interests include social networks, diagram understanding, design of systems, and sensor networks.

Barbara Tversky is a Professor of psychology and education at Teachers College, Columbia University. She attained her BA, MA, and PhD in cognitive psychology at the University of Michigan. Dr. Tversky's research interests are in spatial language and thinking; event perception and cognition; diagram production and comprehension; gesture, diagram, and language in communication; and diagrams for discovery and design.

James E. Corter is a Professor of statistics and education at Teachers College, Columbia University. He attained his BA in psychology (highest honors) in 1977 and his MA from the L. L. Thurstone Psychometric Laboratory in 1979, both from the University of North Carolina; and his PhD in experimental psychology in 1983 from Stanford University. Dr. Corter's research interests are in computational models of human learning and categorization, judgment and decision making, clustering and scaling methods for multivariate data, statistics expertise and probability problem solving, and the evaluation of educational technology innovations.

Jing Ma is a PhD candidate in the Howe School of Technology Management at Stevens Institute of Technology.