In M. Petre & D. Budgen (Eds) Proc. Joint Conf. EASE & PPIG 2003 Pages 473-479

Team coordination through externalised mental imagery

Marian Petre
Faculty of Mathematics and Computing
The Open University
United Kingdom
m.petre@open.ac.uk

Abstract

Fundamental to the effective operation of a design team is the communication and coordination of design models: that the members of the team are all contributing to the same solution. Other work has shown that breakdowns in the accurate sharing of goals are a significant contributor to bugs, delays and design flaws. This paper discusses one mechanism by which teams unify their vision of a solution. It describes how the mental imagery used by a key team member in constructing an abstract solution to a design problem can be externalised and adopted by the rest of the team as a focal image. Examples drawn from in situ observations of actual design practice of a number of computer system design teams are offered. The examples illustrate how the images were introduced, how they were used to coordinate subsequent design discussions, hence how they evolved, and how short-hand references to them were incoporated into the team's 'jargon'.

1. INTRODUCTION: TEAM COORDINATION

It has long been recognised that coordination is fundamental to the effective operation of a design team, e.g.: "Clark and Brennan (1991) argue that common ground is necessary for effective coordination of all joint activities (in groups)." (Flor, 1998) Breakdowns in coordination are a contributor to bugs, delays and design flaws (e.g., Krasner, Curtis & Iscoe, 1987; Guindon, Krasner & Curtis, 1987).

This paper discusses one mechanism of coordination which has been observed to occur naturally in high-performance development teams: the externalisation of one member's mental imagery for adoption by the whole team as a focal image.

Coordination requires that team members have compatible models of the solution – that the communication from one person's internal model through some medium of representation to another person's internal model is effective, conveying meaning accurately. What is known about the relationship between external representations and internal mental models, or intermal mental imagery? A key question is whether personal mental imagery ever becomes public. A follow-on question is whether personal mental imagery would be of any use if it does become public. Some images and imagery, for instance, may be extremely useful to the individual, but by their nature may be very difficult to describe verbally and to use as a shared metaphor, because they are not well suited to reification and shared physical representations (such as diagrams, gestures, physical analogies, etc).

This paper presents examples of how the articulation of the mental imagery used by an individual in solving a problem can introduce the rest of the team to key insights or perspectives, provide a focus for team discussions, aiding communication and collaborative reasoning, and provide a mechanism for calibrating individual understanding against a shared model.

1.1. Co-ordination in software development methodologies

A number of recent software development methodologies aim to address the team coordination issue, often by creating immersive environments of discourse and artefacts which promote regular re-calibration with the other team members and with artefacts of the project. For example, contextual design (Beyer and Holtzblatt, 1998) describes 'living inside' displays of the external representations in order to internalise the model, referring to the displayed artefacts as "public memory and conscience". Contextual design, furthermore, incorporates explicit model coordination among team members as part of the prescribed process.

In another example, extreme programming (Beck, 1999) emphasises the importance of metaphor: the whole team is required to adopt a metaphor embodying the solution model. Again, the metaphor is relied on as a coordination mechanism, so that the team members know they are all working on the same thing. The metaphor is carried into the code, e.g., through naming, and is included in the documentation.

Radical co-location (Teasley, et al., 2000) shares some of the same features. Team members – working on problems requiring novel solutions or involving many highly interactive parts –work together in a shared space (or 'war room') for the duration of a project, giving them ready access to each other and to work objects. Crucially, they tend to use the walls to display design documents and artefacts, which are then visible to the whole team as a developing record of design ideas and history, and hence can easily be viewed, modified, or referred to in design discussion.

1.2. The importance of external representations

All of the examples given above emphasise shared artefacts and shared representations. A number of researchers have portrayed the importance of external representations in design, both to support design reasoning (model building) and as a medium of communication among designers (model sharing and calibration). Schon (1988) describes: "a design is a 'holding environment' for a set of ideas...designers convey meaning via drawings sometimes without articulation: 'you know what this means'". Similarly, Flor and Hutchins (1991) write about the importance of good external representations for effective design and design reasoning. Scaife and Rogers (1996), examining the potential of graphical representations as 'external cognition', highlight the importance of coordination between external representation and internal model, finding that mis-matches between the two account for many problems with visualisations.

1.3. Externalising mental imagery

So what is known about the relationship between external representations and internal mental models, or internal mental imagery? There is widespread anecdotal evidence (e.g., Lammers's interviews of well-known programmers, 1986) that programmers make use of visual mental images and mental simulations when they are designing programs. Experts form detailed conceptual models incorporating abstract entities rather than concrete objects specific to the problem statement (Larkin, 1983). Their models accommodate multiple levels and are rich enough to support mental simulations (Jeffries et al., 1981, Adelson and Soloway, 1985). A previous study (Petre and Blackwell, 1997) elicited mental imagery of ten individual expert programmers during a design task. It identified a number of diverse forms of imagery (including verbal, spatial, visual, auditory elements) and some common characteristics among them.

It seems intuitively obvious that there are times when imagery does become externalised and when the externalisation is useful. Yet we have found little published evidence of effective, direct externalisation of personal mental imagery in software development, apart from introspective justifications for software tool design. This paper reports a form of externalisation which has been observed to occur naturally in high-performance development teams: when an individual's mental image is adopted for team use.

1.4. Example 1: a typical example

One typical example arose in the context of the mental imagery study mentioned above. The expert was thinking about a problem from his own work and articulated an image: "...the way I've organised the fields, the data forms a barrier between two sets of functions...It's kind of like the data forming a wall between them. The concept that I'm visualising is you buy special things that go through a wall, little ways of conducting electrical signals from one side of a wall to another, and you put all your dirty equipment on one side of a wall full of these connectors, and on the other side you have your potentially explosive atmosphere. You can sort of colour these areas...there's a natural progression of the colours. This reinforces the position queues...There's all sorts of other really complex data inter-linkings that stop awful things happening, but they're just infinitely complex knitting in the data. (Of course it's not pure data...most of the stuff called data is functions that access that data.) The other key thing...is this temporal business we're relying on...the program is a single-threaded program that we restrict to only operate on the left or on the right...a hah!...the point is that the connections to the data are only on one side or the other. The way I organise the data is...a vertical structure, and the inter-linkings between data are vertical things...vertical inter-linkings between the data tell me the consistency between the data, so I might end up, say, drawing between the vertically stacked data little operator diagrams..."

After he described the image fully, the expert excused himself and went down the corridor to another team member, to whom he repeated the description, finishing "And that's how we solve it." "The Wall" as it became known, became a focal image for the group.

2. OBSERVATIONAL EVIDENCE

The evidence discussed here is a 'by-product' of other studies: initially, of a study of programmers' mental imagery (Petre and Blackwell, 1997) from which the above example arises; subsequently of a number of other *in situ* observational studies of early design activity. Those studies had other issues as their focus, for example design representations and processes used by multi-disciplinary concurrent engineering teams, representations (including ephemeral ones) used in early ideas capture, group discussions and processes in very early conceptual design, the generation and use of software visualisations. Thus, the core evidence was accumulated opportunistically from five

different software development teams and ten different projects in three different companies over a period of some five years.

2.1. The experts

The experts, from both industry and academia, and from several countries in Europe and North America, share the same general background: all have ten or more years of programming experience; all have experience with large-scale, real-world, real-time, data- and computation-intensive problems; and all are acknowledged by their peers as expert. All are proficient with programming languages in more than one paradigm. The coding language used was not of particular interest in these investigations, but, for the record, a variety of styles was exercised in the examples, using languages including APL, C, C++, HyperCard, Java, common LISP, macro-assembler, Miranda, Prolog, and SQL. Their preferred language was typically C or C++, because of the control it afforded, but the preference did not exclude routine verbal abuse of the language.

2.2. The companies and teams

All were small teams of 3 to 12 members, all included at least one expert programmer of the calibre of 'super designer' (Curtis et al., 1988), and all were in companies where the generation of intellectual property and the anticipation of new markets characterised the company's commercial success. All were high-performance teams: effective intellectual-property-producing teams that tend to produce appropriate products on time, on budget, and running first time. The companies were small, not more than 200-300 employees, although some were autonomous subsidiaries of much larger companies.

2.3. The domains

Most were in large, long-term (1- to 2-year) projects. Often the software was one component of a multi-disciplinary project including computer hardware and other technology. Industries included computer systems, engineering consultancy, professional audio and video, graphics, embedded systems, satellite and aerospace – as well as insurance and telecommunications. Programmers generate between 5 and 10,000 lines of code per compile unit, typically around 200 lines per compile unit, with on the order of 3,000 files per major project.

It is important to note that these experts work in relatively small companies or groups that typically produce their own software rather than working with legacy systems. The software they produce is 'engineering software' rather than, for example, information systems, although products may include massive data handling and database elements. Goel (1995) argues, in the context of external representation, that there is a principled distinction to be made between design and non-design problems. That distinction is pertinent here, and the results presented may not generalise beyond this variety of design and this style of working.

2.4. Limitations

Experts are well-known for rationalising their practice 'on-the-fly'. As reported by Schooler, Ohlsson & Brooks (1993), there is evidence that solving insight problems relies on essentially non-reportable processes, even that verbalisation interferes with some important thought processes. On the other hand, although subjective tests may be suspect, they have in some cases been shown to be reliably consistent, and to produce results just as good as those from more objective tests (Katz, 1983) There is some evidence that self-ratings do correlate with demonstrated ability (Ernest, 1977) and are stable in cases where they do. These studies relied on subjects whose reports of activity in earlier studies corresponded well to other evidence of their activity, such as notes and observed actions, i.e., it relied on subjects who appeared to be 'good self-reporters'.

3. HOW THE EXAMPLES OCCURRED

In the observed examples, the mental imagery used by a key team member in constructing an abstract solution to a design problem was externalised and adopted by the rest of the team as a focal image. The images were used both to convey the proposed solution and to co-ordinate subsequent design discussions. The examples all occurred in the context of design, and the images concerned all or a substantial part of the proposed abstract solution.

3.1. The nature of the images

The images tend to be some form of analogy or metaphor, depicting key structural abstractions. But they can also be 'perspective' images: 'if we look at it like this, from this angle, it fits together like this' — a visualisation of priorities, of key information flows or of key entities in relationship. The image is a conceptual configuration which may or may not have any direct correlation to eventual system configuration.

Typically, the image embodies a major insight in the solution of the problem: it identifies which model underpins the solution. In doing so, it often also embodies a major insight into which problem – or which interpretation of the problem – is being solved. The images themselves sometimes appear 'obvious', known solutions to familiar problems – but their effect on the project is profound, because of the insight step that made the solution evident.

3.2. The process of assimilation

In all of the examples observed, the image was initially described to other members of the team by the originator. Members of the team discussed the image, with rounds of 'is it like this' in order to establish and check their understanding. Although initial questions about the image were inevitably answered by the originator, the locus did shift, with later questions being answered by various members of the team as they assimilated the image. The image was 'interrogated', for example establishing its boundaries with questions about 'how is it different from this'; considering consequences with questions like 'if it's like this, does it mean it also does that?'; assessing its adequacy with questions about how it solved key problems; and seeking its power with questions about what insights it could offer about particular issues. In the course of the discussion and interrogation, the image might be embellished – or abandoned.

3.3. They are sketched

Sketching is a typical part of the process of assimilation, embodying the transition from 'mental image' to 'external representation'. The sketches may be various, with more than one sketch per image, but a characteristic of a successful focal image is that the 'mature' sketches of it are useful and meaningful to all members of the group. This fits well with the literature about the importance of good external representations in design reasoning (e.g., Flor and Hutchins, 1991; Schon, 1988; and others).

3.4. Continuing role reflected in team language

If the image is adopted by the team, it becomes a focal point of design discussions, and key terms or phrases relating to it become common. Short-hand references to the image are incorporated into the team's jargon to stand for the whole concept. But (unlike the metaphors used in extreme programming) the image is 'team-private'; it typically does not get passed outside the team and typically does not reach the documentation.

4. EXAMPLE 2: COORDINATING RE-THINKING

In a project involving scrolling through a file, the engineers had previous experience of systems in which a file could be played through at any speed (including zero), but only forward. Such a system involved a pipeline of processes linked by queues. Each process acted as consumer for the preceding process and producer for the following process with the intervening queues (FIFOs) matching the data rates.

When asked to produce a system which could be scrolled through either forward or backward, their first solution built on the existing model: flushing all the queues and all the processes and then re-initialising everything in the opposite direction. Although this worked, there was a considerable delay at changes of direction.

One engineer suggested viewing the problem "as an unrolled movie film laying across the table, not as I watch it on the screen". He explained that he saw the process as a positional device (trying to go to a position along a permanently available array of data) instead of as a temporal device (going slower or faster through a timed stream of inputs, each of which is processed as it is presented). As soon as the team had grasped this image, all the appropriate data structures were agreed in minutes, and questions about how to link processes were resolved by reference to the "movie on the table" model.

As reflected on by another team member: "All those FIFOs had us in a right mess when we did reverse. Seeing it (the stream of data) as flat meant you could walk along it in either direction. When we only saw it against time, then going backwards was like time travel, you just can't do that, but going left and right? Hey that's easy."

This example reflects the role of externalised mental imagery in helping designers 'out of the box' of familiar thinking and into a reassessment of the fundamental problem, rather than of the problem as interpreted by previous solutions. In this case, the team started from the previous solution; the insight step embodied in the externalised image resulted from a re-reflection on the problem, and the sharing of the image drew the rest of the team into the revisiting the problem interpretation.

5. EXAMPLE 3: THE INDIVIDUAL'S MENTAL IMAGE OF THE PROBLEM PROVIDES LEVERAGE FOR OTHERS' INSIGHT INTO THE SOLUTION

In a 'lossy' data compression problem (similar to a jpeg encoder) the traditional method had been to design a filter to model the psycho-perceptual error detection ability of the human evaluator, and then to use this filter to shape the error inevitably produced by the data compression, so as to make it less perceptible. The effect of this method was to produce compression software which tried to "guess" the correct decision to make at a particular stage and, having made it, then looked at further input and generated further output.

One engineer suggested that he saw the problem as "like a chess game, but like playing chess without evaluating my own move in the light of what the opponent would do

next". When questioned by other members of the team, he suggested that the viewer's eye was an opponent in a guessing game. "Once you see this thing as a game between us (the encoder software) and the viewer (the error evaluation function), it's obvious that we can't possibly win unless we search a tree of possible answers, not just accept our best guess right now and charge blindly on." As soon as this "game playing" view was adopted by the team, after a period of interrogation and discussion, they incorporated "look-ahead" into their solution, importing tree-pruning and scoring functions.

Reflected another member of the team: "I really can't see why we were so dumb not to see this as a game-type thing months ago, I guess we were all tied up in the real-time-ness of the thing we never thought about look-ahead or multiple candidate paths."

An interesting characteristic of this example is that the focal mental image provided a model of the *problem* – not of the existent solution. The image evolved through discussion and interrogation, clarifying roles and goals, and exposing implications to all the team members. The re-design of the solution arose from the insights provided by the new view on the problem.

6. EXAMPLE 4: DIFFERENT MODELS OBSTRUCT COLLABORATION

Two sub-teams on a project involving replaying sound files from a computer disk held two essentially different images of how this process should be approached, and in particular how to deal with the simultaneous replaying of multiple files. Each sub-project leader developed – and shared – an image based on an existing piece of mechanical sound equipment they thought they were modelling. One sub-team adopted the model of a disk-based sound editor, which supposed paired files and allowed only cross-fades between two files, and then summed multiple pairs. The other sub-team adopted the model of a multi-track tape recorder and mixer, which supposed that any number of files could be replayed with potentially multiple overlaps. Each sub-team coordinated effectively around its own shared image.

As the project progressed, each sub-team developed private names for the applicable objects, based on their focal image. Unfortunately there were only a limited number of names in common use to describe pieces of a sound file, and several words, 'channel' and 'track' in particular, were used by both groups – but to mean different, and conflicting, things. In meetings between the two sub-projects, each with its own fiercely-held private image/model, the arguments about whose model was 'right' prevented any cooperation.

Unfortunately the incompatibility between the two models reached through every level of their hardware, firmware and software. In the end the only place where the two sub-projects could be connected was at their top-level interfaces; any attempt to link them at a lower level foundered on the incompatibility of their models. Of course both models were correct and worked fine – just not together.

Team members reflected on the effect of the disparity: "In the end I guess both groups realised that either model would act as a hook to hang all our architectural decisions on, but using both really screwed things up good." And: "The tracks per channel thing came straight out of what you erected as your model for the world. An editor gave you two, a multi-track gave you many. Life was easy as long as you stayed in your own area, looking at what the other guys were doing made your head hurt."

Each sub-team followed a typical pattern in their adoption of a focal image from an individual's mental imagery, in each case a fairly obvious mechanical analogy. But the example also provides a counter-example, showing how the adherence to the sub-team model and language prevented integration between the two parts of the project.

7. DISCUSSION

The features observed in expert practitioner behaviour in this domain are consistent with findings in a range of related literatures (such as mental imagery, problem solving, memory and schema theory).

The images discussed and interrogated by the team provide a co-ordination mechanism. Effective co-ordination will by definition require the use of images which are meaningful to the members of the group. The literature on schemata provides explanation here (e.g., Bartlett, 1932). Co-ordination – meaningful discourse – requires shared referents. If there is a shared, socially-agreed schema or entity, this can be named and called into play. But what happens when the discourse concerns an invention, an innovation, something for which there is no existing terminology, no pre-existing schema? A preverbal image in the head of one participant, if it cannot be articulated or named, is not available to the group for inspection and discussion. The use of extended metaphor, with properties in several different facets, provides a way of establishing a new schema. The borrower chooses what is salient in the properties of interest. In describing the image, the borrower is establishing common reference points, co-ordinating with the rest of the team a shared semantics (cf. Shadbolt's research (1984) on people's use of maps and the establishment of a common semantics). The discussion of the metaphor allows the team to establish whether they understand the same thing as each other. The establishment of a richly visualised, shared image (and the

adoption of economical short-hand references) facilitate keeping the solution in working memory (e.g. Logie, 1989).

Schemata may be of varying levels of complexity and abstraction; their importance is in providing structure and economy. Chi et al. (1988) suggest that the nature of expertise is due largely to the possession of schemata that guide perception and problem solving – i.e., experts have more and better schemata than novices. Simon (1973) observes that, when a task is ill-defined, users resort to pre-existing concepts: stereotypes, schemata, or other knowledge. Cole and Kuhlthau (2000) see the use of schemata as fundamental to sense-making at the outset of problem solving: the problem-solver invokes a schema or model of the problem in order to create a frame of reference and hence to identify the initial problem state.

On one hand, the use of existing schemata enables the user to take some action in unfamiliar or ill-defined tasks. On the other hand, the use of existing schemata can lead to misconception, mis-action, or fixedness. (Tourangeau and Sternberg, 1982) This is illustrated in examples 3 and 4.

The examples have indicated both the power of an effective externalised image -- particularly imagery that embodies an insight or a distinctive perspective – and the danger of discrepant or limited images. In high-performance teams, the interrogation of shared images tends to expose inadequacies. Shared images are often discarded, although they nevertheless assist the design process by supporting discussion and cooperative reflection.

The implication may be that providing access to a wider range of source imagery, having different properties, might support this process. This is consistent with the observation from the expertise literature that experts remember large numbers of examples – indeed, the literature suggests that experiencing large numbers of examples is a prerequisite to expertise (e.g., Chi et al., 1988). The typical expert ability to detect resonances across domains can be applied to detect links between abstracted examples, i.e., to identify useful metaphors.

8. CONCLUSION

It appears that, in the context of the design and generation of 'engineering software', the externalisation of expert mental imagery can play an important role in the design reasoning. Individual imagery does sometimes enter external interaction in a way that is useful. The mental imagery used by a key team member in constructing an abstract solution to a design problem can in some cases be externalised and adopted by the rest of the team as a focal image. A key to this is that the individual (typically, but not always an expert) is able to articulate his own mental imagery in a way that frames the problem and/or solution for others in the team. Discussing, sketching and 'interrogating' the image helps the team to share the insight and to co-ordinate their design models so that they are all working on the same problem – which is fundamental to the effective operation of the team.

ACKNOWLEDGEMENTS

The author is profoundly grateful to the expert programmers, without whom the paper would not be possible, and to their companies which permitted access. Thanks are due to colleagues who provided essential commentary, including Alan Blackwell, Peter Eastty, Marc Eisenstadt, Henrik Gedenryd, Simon Holland, William Kentish, Jennifer Rode, and Helen Sharp. Special thanks are due to Gordon Rugg, who was instrumental in writing the paper. Some of the observations were conducted under EPSRC grant GR/J48689 (Facilitating Communication across Domains of Engineering). Others were conducted under an EPSRC Advanced Research Fellowship AF/98/0597.

REFERENCES

Adelson, B., and Soloway, E. (1985) The role of domain experience in software design. *IEEE Transactions on Software Engineering*, **SE-11** (11), 1351-1360.

Bartlett, F.C. (1927) The relevance of visual imagery to thinking. *British Journal of Psychology*, **18** (1), 23-29.

Bartlett, F.C. (1932) *Remembering: An Experimental and Social Study*. Cambridge University Press.

Beck, K. (1999) *Extreme Programming Explained: Embrace Change*. Addison-Wesley.

Beyer, H., and Holtzblatt, K. (1998) Contextual Design: Defining Customer-Centered Systems. Morgan Kaufmann.

Chi, M.T.H., Glaser, R., and Farr, M.J. (Eds) (1988) *The Nature of Expertise*. Lawrence Erlbaum.

Cole, C., and Kuhlthau, C.C. (2000) Information and information seeking of novice versus expert lawyers: how experts add value. *The New Review of Information Behaviour Research 2000*. 103 – 115.

Curtis, B., Krasner, H., and Iscoe, N. (1988) A field study of the design process for large systems. *Communications of the ACM*, **31** (11), 1268-1287.

Ernest, C.H. (1977) Imagery ability and cognition: a critical review. *Journal of Mental Imagery*, **1** (2), 181-216.

Flor, N.V. (1998) Side-by-side collaboration: a case study. *International Journal of Human-Computer Studies*, **49**, 201-222.

Flor, N.V., and Hutchins, E.L. (1991) Analysing distributed cognition in software teams: a case study of team programming during perfective software maintenance. In: J. Koenemann-Belliveau, T.G. Moher and S.P. Roberston (Eds), *Empirical Studies of Programmers: Fourth Workshop*. Ablex.

Goel, V. (1995) *Sketches of Thought*. MIT Press.

Guindon, R., Krasner, H., Curtis, B. (1987) Breakdowns and processes during the early activities of software design by professionals. *Empirical Studies of Programmers: Second Workshop*. Ablex., 65-82

Jeffries, R., Turner, A.A., Polson, P.G., and Atwood, M.E. (1981) The processes involved in designing software. In: J.R. Anderson (Ed) *Cognitive Skills and Their Acquisition*. Lawrence Erlbaum. 255-283.

Krasner, H., Curtis, B., and Iscoe, N. (1987) Communication breakdowns and boundary spanning activities on large programming projects. *Empirical Studies of Programmers: Second Workshop*. Ablex. 47-82

Lammers, S. (1986) Programmers at Work. Microsoft Press.

Larkin, J.H. (1983) The role of problem representation in physics. In: D. Gentner and A.L. Stevens (Eds), *Mental Models*. Lawrence Erlbaum.

Katz, A.N. (1983) What does it mean to be a high imager? In: J.C. Yuille (ed), *Imagery, Memory and Cognition: Essays in Honor of Allan Paivio*. Erlbaum.

Logie, R.H. (1989) characteristics of visual short-term memory. *European Journal of Cognitive Psychology*, **1**, 275-284.

Petre, M., and Blackwell, A. (1997) A glimpse of programmers' mental imagery. In: S. Wiedenbeck and J. Scholtz (Eds.), *Empirical Studies of Programmers: Seventh Workshop*. ACM Press. 109-123.

Scaife, M., and Rogers, Y. (1996) External cognition: how do graphical representations work? International Journal of Human-Computer Studies, **45**, 185-213.

Schon, D. (1988) Design rules, types and worlds. *Design Studies*, **9** (3), 181-190.

Schooler, J.W., Ohlsson, S., and Brooks, K. (1993) Thoughts beyond words: when language overshadows insight. Jo*urnal of Experimental Psychology: General*, **122** (2), 166-183.

Shadbolt, N.R. (1984) Constituting Reference in Natural Language: *The Problem of Referential Opacity*. PhD Thesis, University of Edinburgh.

Simon, H.A. (1973) The structure of ill-structured problems. *Artificial Intelligence*, **4**, 181-202.

Teasley, S., Covi, L., Krishnan, M., and Olson, J. (2000) How does radical collocation help a team succeed? CSCW 2000 (Philadelphia, Dec. 2-6). ACM Press. 339-346.

Tourangeau, R., and Sternberg, R. (1982) Understanding and appreciating metaphors. *Cognition*, *11*, 203-244.