Creative Expertise and Collaborative Technology Design

Linda Candy[1] and Ernest Edmonds[2]

[1]Key Centre of Design Computing and Cognition
Faculty of Architecture, Design Science and Planning
University of Sydney, NSW 2006, Australia
[2] Creativity and Cognition Studios, Faculty of Information Technology
University of Technology, Sydney, NSW 2007, Australia

Abstract. The paper is concerned with increasing our understanding of creative expertise drawing upon studies of collaboration between technologists and artists. The nature of expertise in collaborative creative work is discussed and the implications for support tools considered. Characteristics of a visual programming environment used to support collaboration in the development of interactive digital works are discussed. Such environments provide facilities for sharing representations between experts from different domains and in this way can be used to enable all parties to the collaboration to participate fully in the decision making process.

1 Introduction

Creative people with different areas of expertise frequently work together in professional practice. Enterprises such as film-making and building design and construction require groups of experts to collaborate in the creation of new ideas and artifacts towards the realization of the overall projects. This paper is concerned with increasing our understanding of both creativity and expertise drawing upon studies of creative collaboration between experts from different domains. Creativity is characterized as the invention of new forms that have not been represented or realized in any explicit way. We define *creative expertise* as the application of expertise to previously unidentified problems and the creation of innovative solutions to meet novel situations, in this case interactive digital artworks. We may say that someone possesses creative expertise when they are outstandingly good at finding new forms in a specific domain: for example, an artist might have creative expertise in painting or a software engineer might have creative expertise in object-oriented systems. The paper addresses two main issues: first, the characteristics of collaborative creativity between experts and second, the role of technologies in facilitating successful creative collaboration. In particular, we identify issues about Human-Computer Interaction (HCI) requirements of the software used in such work.

In the studies referred to here, small groups of experts from different domains worked together towards the realization of creative projects defined by artists. A specific focus of the paper is the role of technologists collaborating with artists in the creation of interactive digital works. The term *technologist* is used to designate the

role of designer and implementer of the interactive tools, systems and installations. In the studies described, all the technologists were selected for their expertise in the development and application of various digital hardware and software systems for creative art, music and design. Each person also had a specific area of expertise that was essential for the particular project in which he or she was the primary collaborating partner: for example, expertise in sound analysis software or visual programming languages. The studies took place in the COSTART project (Candy and Edmonds, 2002a). Collaborations between technologists and artists were established initially on the basis that the technologist's role was to provide solutions to specific problems arising out of a proposal determined by the artist partner. Experts with specific skills areas were identified and matched to the people and their projects. It was discovered that the characteristics of experts best suited to working collaboratively in creative contexts extended well beyond having deep knowledge of the field, in this case digital technology. The expert as "solution finder", whilst highly appropriate for traditional consultation and advisory roles, has its limits when he or she ventures into the creative space as a "problem finder" with altogether different demands upon professional expertise. Whether collaborating about the design of an artefact or about a computer program to control an environment, the design intentions and options have to be made explicit and then shared amongst the team. Identifying acceptable ways of doing this is important. In the cases under consideration, software development was at the heart of each project: the representation used for that software and how it contributed to the sharing of ideas was critical to the progress of the work.

2 Practice-based Research in Collaborative Creativity

In order to understand more fully the issues discussed above, a new approach to the creation of technology-based art was developed. Practice-based research involves two complementary and interdependent processes: innovative technology-based art projects are developed in tandem with observational research activities. The aim of the research is to acquire information that guides change in the existing situation but also to increase our understanding of creative practice and the requirements of future digital technologies. The aim of the practice, on the other hand, is to create new forms of art and technology systems for exhibition in galleries and public places as ends in themselves. The approach involves forming collaborative teams, gathering research data, collating and analyzing the results and disseminating knowledge on the basis of the evidence. The methods used are based upon ethnographic approaches and user-centered techniques. The research aims were to identify the key activities of the creative process and to understand the nature of collaboration in design and development of interactive technology. The artist residency, a familiar experience in the artistic community, is at the heart of the process. However, it is a residency that differs in a number of important respects from the conventional type. In order to maximize the opportunities for acquiring reliable evidence, the data gathering process must be transparent to the research team, all of whose members have significant roles in assessing the feasibility of the prospective studies, facilitating the conduct of the residencies and collecting

primary data. The preparation of material is shared across the team and resulting information distributed where it is relevant to decision making. All participants, artists, technologists and observers, kept a daily record of events as they happened. Images and prototypes of work in progress were kept for reference and illustration. Experiences were recorded about what was proposed, discussed, carried through, what stumbling blocks arose, how they were addressed. Perceptions as to whether the ideas were workable, interesting, challenging were noted and whether the collaboration worked well or not. Reflections about whether the technical solutions worked well, or not, were recorded at the time and in follow up interviews and meetings. The many types of data about the activities and outcomes of the art-technology collaborations were assessed. Diaries kept by artists, technologists and observers were collated into a single transcription record for each case. Transcriptions of key meetings and the final interview were documented in a data repository as sound files and text records. The data was compiled and structured in chronologically ordered transcription records for each case. This provided the primary evidence for the extraction of features and descriptors of collaboration and was carried out by different analysts in order to arrive at independent viewpoints. The approach is described in Candy and Edmonds, 2002a.

3 Expertise, Creativity and Collaboration

When experts collaborate, the incentive to do so may spring from different motivations. In many organizations, the expert's role is to provide a specialized contribution to a complex project in which there may be several other contributing disciplines. Experts identify potential problem areas and bring forward solutions to these in advance of the physical realization of the product. Being an expert implies having special knowledge that confers the status of an authority within a particular community of experts as well as the wider public. The expert is most often called upon to provide insight and analysis based upon an ability to draw rapidly from up to date and pertinent information that can provide appropriate solutions to specified problems. The expert is a specialist rather than a generalist and as such the expertise is based upon recognized strategies, skills and methods that have been developed and honed with experience. In the review by Glaser and Chi (1988), expertise is perceived to be domain specific and whilst experts are good at perceiving large patterns in their own domain and applying the knowledge very quickly, this is for routine work only. Hewett relates some of the known characteristics of expertise to that of the artist embarking on a digital work (Hewett, 2002). He points out that technology expertise involves finding efficient solutions to a given problem and this focus does not easily lend itself to the kind of open exploration that is important in creative practice.

When people collaborate, there is usually a significant degree of self-interest involved: there is a need for something they cannot supply themselves. If the collaborative project is complex and involves difficult tasks, it makes sense to collaborate with someone who possesses complementary knowledge and skills. In that sense, the very basis of collaboration is often *difference*: whilst the parties may both be expert in their own right, the expertise itself comes from different disciplines. Where the context is a

creative one, the nature of the creative vision and aspirations play an important role in defining and shaping the role of the expert. Bringing expertise into a collaborative situation involves a process that is different to one of commissioning expertise as a specific contribution to a problem situation. Expertise as a consultative process may take the form of writing a report based on a single visit or series of visits which is then used by the commissioning parties to address the problems identified. Expertise in collaboration, however, involves the development of sustainable relationships between the participating parties as well as the contribution of domain knowledge and skill. In addition, expertise applied in creative contexts requires a degree of motivation and commitment that is intrinsic to the activity, as distinct from being driven by extrinsic factors such as financial incentives. Nickerson's recommendations for improving creativity could be applied equally to expertise (Nickerson, 1999). These are summarized as follows:

- Establish Purpose and Intention
- Build Basic Skills
- Encourage Acquisition of Domain-specific Knowledge
- Stimulate and Reward Curiosity and Exploration
- Build Motivation
- Encourage Confidence and Risk Taking
- Focus on Mastery and Self-Competition
- Promote Supportable Beliefs
- Provide Balance
- Provide Opportunities for Choice and Discovery
- Develop Self Management (Meta-Cognitive Skills)
- Teach Techniques and Strategies for Facilitating Creative Performance

3.1 Creative Expertise and Collaboration

In collaborative work, expert skills are essential contributions to the process and provide the basis for the choice of partners. However, where the work is also creative, those specialised skills, whilst essential for certain tasks, are not enough unless they are combined with other attributes. Being an expert in the conventional sense of the term, i.e. a specialist who has significant levels of knowledge about a well-defined domain, could be a negative thing for creative collaborative work if it operates in such a way as to inhibit contributions from the team. Where the project is of a creative and high-risk kind, expertise that is used to provide answers and solutions to given problems, must be combined with other kinds of personal characteristics. From the studies undertaken, we have identified a set of individual characteristics and other collaboration elements that are necessary for the effective conduct of collaborative creative projects (Candy and Edmonds, 2002b).

In the follow up studies referred to here, the technology experts were selected for their knowledge of digital technology that was applicable to creative work. The range of areas of expertise included design support systems, music analysis and production tools, sound and vision systems as well as a number of programming languages. They

also had foundation skills ranging across a number of different disciplines: music, graphic design, product design, business etc. from which they had developed their interests in using and developing digital tools for creative purposes. Educational experience was diverse and no single subject discipline dominated. With employment and further education all were involved in migrating existing knowledge into new areas. Examples of issues that were identified are summarized below.

In one case, both collaborating parties had a high degree of background education, knowledge and experience in common and both were qualified in different areas of music with a mutual interest in electronic music. The technologist was also formally trained in classical music and was able to link music and technology through the skills developed in her music degree.

It was an important selection criterion that the technology collaborators were able to demonstrate openness about artistic work and, therefore, evidence of personal creativity was an important factor. The person who was, on the face of it, the least involved in creative work initially adopted a supportive but artistically detached, approach. That position proved to be hard to sustain and changed as the work developed and the technical challenges grew.

The processes of collaboration depended on the artist as well as the technologist and in some cases that encouraged more creativity on the part of the technologist than others. In one case, it was noted that the technologist felt concerned that his own creativity might be seen as interference in the artist's process. It was not clear, however, that the artist necessarily agreed. So in that sense, an opportunity for personal creativity in a collaborative team can be quite a complex issue.

It proved to be important to ongoing commitment for the technologists to be engaged in something they respected and enjoyed doing. All had different initial reactions to the artistic work itself. In one case, there was an explicit wish to have active engagement with the conceptual basis for the work from the start. In the other two cases, there was a growth of interest during the development process and a wish to be engaged with the ideas behind the work as well as its technical implementation. Where this was achieved there was greater commitment to the fulfillment of the project goals. In a certain sense, this implies a need for "ownership" of the project.

3.2 Creative Expertise and Learning

The ability to learn new techniques in very short time scales was important. The basis for that learning rested in existing skills that were readily transferable. Learning how to use a programming environment such as Visual Basic competently in a matter of two or three days was relatively easy for someone who already knew the C and C++ programming languages. This also applied to knowledge of software applications: for example, using two types of music analysis software could be combined quickly to support experimentation with ways of developing the synthetic language that was the main goal of the artist. These transferable skills were useful in finding practical ways of moving forward in order to progress difficult and ambitious goals. Being able to offer such skills facilitated exploration and discovery in the creative process.

A striking concern that is shared by all of the three technologists discussed here is having an opportunity to learn how to do something that has not been done before. Learning is a central motivator and projects that do not require it tend not to be very stimulating. It is not the case that having high levels of expertise precludes the need to learn something new. In the cases discussed here, the collaborative experience was often explicitly used to generate new problems or situations that brought with them the need to learn a new technique. Sometimes it was even the case that the unexpected, interventions of others, actually stimulated what was seen as the most interesting aspect of the work.

"…but the other interesting thing is that in Athens, without my knowledge the people who organized it put a video camera behind my head and projected it on my screen. And I thought – I didn't know this and I thought I'd played a really rubbish set I was really unhappy with it, the sound was terrible, in fact it did sound terrible it was a lot worse than anywhere else I'd played but the audience loved it, people even came round and said wow that was brilliant and it was because they could see the screen and they could see what I could see which was all the patterns shifting and all the things changing and I had a visual kind of representation of it and which I thought was interesting"

"Yes, so I'm hoping there'll be some new aspect to it.
I'm quite keen to learn something new as well.
You find out something new that's the exciting bit "

The unknown, the new and the challenging tasks seem to be the ones most likely to motivate the expert. In the cases reviewed here, it was the artist's tendency to generate such problems that often gave the collaboration its life and interest to the technologist. Learning new things is inherently interesting to the expert despite their presumed significant existing knowledge.

4 Collaborative Technology Development

In a majority of the COSTART case studies, new software was designed, implemented and evaluated as a collaborative activity between technologists and artists. Each residency project gave rise to an interactive artwork, performance or installation and in five cases, the Max/MSP visual programming language provided the basic software implementation environment (cycling74). The use of the Max/MSP environment enabled the artists, although often not expert programmers, to take an active role in the development of the systems alongside the technologists.

One issue for shared representations arises from the desirable practice of generating software prototypes that the user (in this case the artist) can evaluate. The prototype is typically not something that can or does evolve into the delivered system. It is built in a fast development environment that does not attempt to offer all of the functions or performance desired. Instead, it allows something that looks rather like the intended

end result to be made quickly – and then thrown away. The issue is to ensure that a good looking prototype does not lead the user to believe that the work is largely completed when, in point of fact, it has hardly started. One approach is to use an evolutionary approach by working in software development environment that allows rapid change and also provides easy to read representations of the code.

The origin of Max/MSP is in electronic music studios and the forms used are quite easy to understand for musicians who have worked in that context. It turns out that artists working in visual interaction sometimes also find Max/MSP quite understandable without much training (Edmonds et al, 2003).

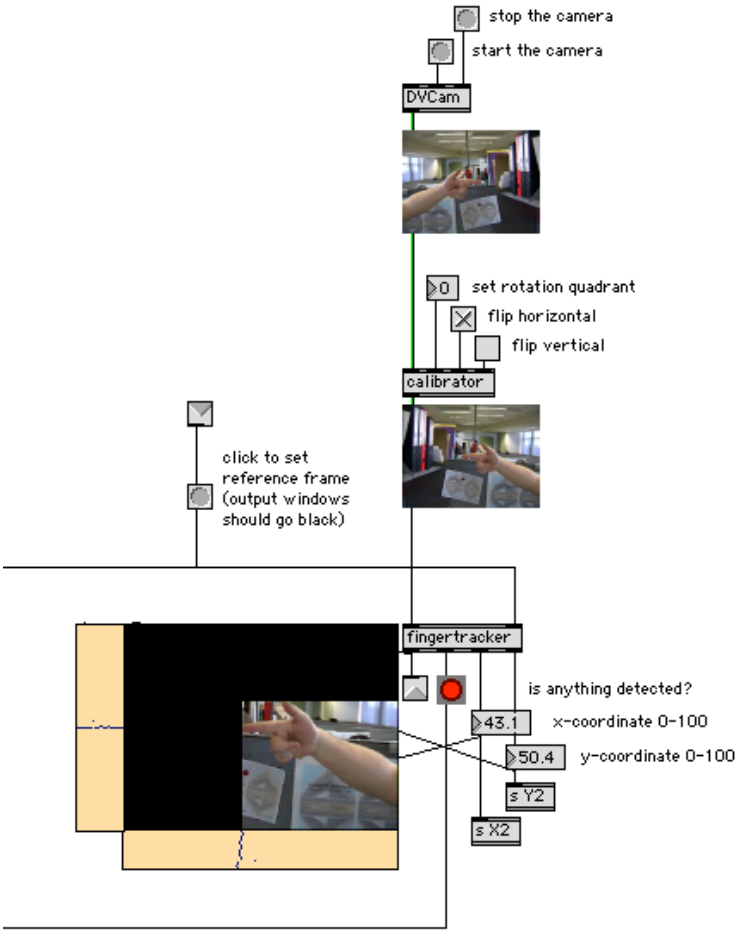

Fig 1. Part of the Max code for one of the artworks

When working with Max/MSP the process of software development is not defined by a linear sequence of distinct stages. Where particular stages can be observed, they are largely determined by separate technological processes. These, in turn, generate software tasks that can be developed in parallel. We also observed that, as the process of making art changed, there was a shift in the allocation of shared responsibility in that the artist was able to engage with technical processes and the technologist was able to contribute to previously aesthetic or artistic content. This sharing of knowledge and crossover of skills is one that can lead to innovation in both art and technology.

4.1 Characteristics of Max/MSP

Max/MSP is a programming system that is largely developed by its user community. It provides an open-ended framework for developing new objects that can be written in the C language. There are several communities of artists and programmers developing new objects, and, as new technologies and approaches to interaction arise, new objects find their way into these communities. It is in this way that Max both 'keeps up with' and contributes to a dialogue between the needs and interests of the artist-programmers and their tools. External and third party objects provide increasingly simple ways of doing complex tasks. For example, what might have been done with several more basic objects, can be achieved with one custom object. The developmental process is not only streamlined, but also the general vocabulary of the user community is extended. As is the case with open system environments, the user interface to this programming environment is largely developed by the user community itself.

Max/MSP provides the facility to integrate low-level system inputs with high-level visual programming tools so that a single software system can be used for the entire process. In the projects studied here, it facilitated the easy integration of separately developed components. In one residency project, the technologist concentrated on developing a system for aural manipulation while the artist concentrated on visual manipulation. When the whole system needed to be evaluated, these two components were easily linked by links drawn on the screen that defined a flow between the two sections. The graphical interface to the programming system was an important contributing factor to the projects.

Max/MSP not only supports but also encourages different ways of approaching technology-based art. It conforms to the needs of the artistic community where different and contradictory strategies are often nurtured. This attitude is especially evident in contemporary practices where emphasis is firmly placed on process as opposed to artifact. In one project, it became clear that Max was able to promote radical methods. Both artist and technologist felt that it removed the constraints of working within the 'top down versus bottom up' approach to making technological art, and in place of this opposition asserted a heterogeneous collection of methods. This is in contrast to a standard software design life cycle in which the user provides the client requirements definition which is carried out by the software development team, the results of which are then evaluated by the user, feedback given and a new version created in response to

that feedback. Thus a much more intimate communication and highly iterative user-developer relationship is enabled by this kind of approach.

4.2 Shared Code Representations for Collaboration

In the cases described, Max/MSP, with its graphical representation, was seen to be helpful as a shared form to facilitate the collaboration. The software being developed could be discussed and considered in itself and its definition as well as in terms of simply what it put into effect. However, there is a disadvantage to the use of such shared representations. One of the technologists did not find Max/MSP a representation that gave him sufficient information about the details of what the computer was going to do. That was offered in a more acceptable way by languages such as C or Java. These languages, on the other hand, are quite inappropriate for using as shared representations in multi-disciplinary teams as we find here. Thus we see a tension between the preferred shared representation and the preferred technical representation. One answer, not explored here or more generally to any great extent, at this time, is to facilitate multiple views of the same code. One example that is somewhat in this direction and can be sited is the alternate views of html code in web development environments, such as Dreamweaver, where the user can switch between looking at the web page design, as it will appear to the user, and the code that generates it (Macromedia Dreamweaver). They may also see both views side by side.

During one project, a number of systems that were only obliquely related to his proposal were investigated. It was the discussions that ensued, though, which led to a clear understanding between both parties about, on the one hand, what was required and, on the other hand, what sort of things could be done. A note from the technologist's diary illustrates the process:

> *"…after I showed him the ID Studiolab colour database thing we came back to that- a discussion of the people in a orchestra moving about according to some rules- when one moves that causes the others to move in response.*
> *In this case, it was possible to implement some parts of the newly emerging concepts as the week progressed and this helped the two parties to develop a shared understanding of, and a shared language for describing, the problem at hand."*

The need for a shared language in support of collaboration has a very particular implication for complex projects that include significant software development. For the technologist, the implementation language is important and it, or something close to it, is treated as if it was also the description of the design of the system. The team, however, may need the code to be represented from different viewpoints in different notations for the different collaborators.

5 Conclusions

In the increasingly important area of technological expert involvement in creative projects we have seen that there are a number of important human-computer interaction issues that need to be considered. These issues arise primarily from the need to share ideas and potential problem solutions amongst a multi-disciplinary team. The paper has reviewed the issues on the basis of a set of studies of art projects conducted under experimental conditions and has pointed to a way forward in the development of support systems appropriate to such work. Characteristics of the Max/MSP visual programming environment used to support collaboration in the development of interactive digital works were discussed. Such environments provide facilities for sharing representations between experts from different domains and in this way can be used to enable all parties to the collaboration to participate fully in the decision making process.

Acknowledgements

The authors are indebted to the efforts, reflections and comments of the technologists and artists who participated in the second phase of the COSTART project. The work was partly funded by the UK Science and Engineering Research Council.

References

1. Candy, L. and Edmonds, E.A.: Explorations in Art and Technology, Springer Verlag, London (2002a)
2. Candy, L. and Edmonds, E.A.: Modeling Co-Creativity in Art and Technology, In Hewett, T. T. and Kavanagh, T. (eds) Proceedings of the Fourth International Conference on Creativity and Cognition, ACM press: New York (2002b) 134-141
3. Crabtree, A.: Designing Collaborative Systems: A Practical Guide to Ethnography, Springer-Verlag, London Ltd (2003)
4. Cycling74 Max/MSP: http://www.cycling74.com
5. Edmonds, E. A., Candy, L., Fell, M., Knott, R. and Weakley, A.: Macaroni Synthesis: A Creative Multimedia Collaboration. In Banissi, E. et al (eds) Proceedings of Information Visualization 2003, IEEE Computer Society, Los Alamitos, CA. (2003) 646-651
6. Hewett, T.T.: An Observer's Reflections: The Artist Considered as Expert. In Candy and Edmonds, Explorations in Art and Technology Ch13, (2002) 137-144
7. Glaser, R and Chi, M.T. H.: Overview, In Chi, M.T.H., Glaser, R. and and Farr, (eds). The Nature of Expertise. Erlbaum, Hillsdale, NJ (1988)
8. Macromedia Director: http://www.macromedia.com/software/director/
9. Macromedia Dreamweaver: http://www.macromedia.com/software/dreamweaver/
10. Nickerson, R.S.: Enhancing Creativity, In R.J. Sternberg (ed). Handbook of Creativity, Cambridge University Press, Cambridge, UK, (1999) 392-430.