From Isolation to Involvement: Adapting Machine Creativity Software to Support Human-Computer Co-Creation

Anna Kantosalo, Jukka M. Toivanen, Ping Xiao, Hannu Toivonen
Department of Computer Science and Helsinki Institute for Information Technology HIIT
University of Helsinki, Finland
anna.kantosalo@helsinki.fi, jukka.toivanen@cs.helsinki.fi, ping.xiao@helsinki.fi, hannu.toivonen@cs.helsinki.fi

Abstract

This paper investigates how to transform machine creativity systems into interactive tools that support human-computer co-creation. We use three case studies to identify common issues in this transformation, under the perspective of User-Centered Design. We also analyse the interactivity and creative behavior of the three platforms in terms of Wiggins' formalization of creativity as a search. We arrive at the conclusion that adapting creative software for supporting human-computer co-creation requires redesigning some major aspects of the software, which guides our on-going project of building an interactive poetry composition tool.

Introduction

Machine creativity and *support for human creativity* are two complementary goals of computational creativity research. The role of the machine in supporting human creativity has been classified by Lubart (2005) into four categories: computer as a managment aid, computer as a communication enabler, computer as a creativity enhancer, and computer as a co-creator in the creative act. It is easy to see how advancements in machine creativity systems could support the role of the computer as a creativity enhancer, or even as a co-creator: A creative system in a certain domain, say poetry, could be used as a creative assistant for a human poet, producing draft poems that the poet could use as inspiration or raw material. This relationship could be taken even further to create a real partnership in which the computer and the user could take turns writing and editing a jointly authored poem.

Such co-creative systems have great potential for transforming the lives of professionals and laymen alike by increasing their creative potential. To aid the development of future co-creative systems and their integration to everyday lives of people, it is important to gather and analyse knowledge on the design and use of existing co-creative systems.

We use the term human-computer co-creation to refer to collaborative creativity where both the human and the computer take creative responsibility for the generation of a creative artefact. The term co-creation refers here to a social creativity process "leading to the emergence and sharing of creative activities and meaning in a socio-technical environment" (Fischer et al. 2005), but with the emphasis that the computer is, instead of only providing the socio-technical environment, also an active participant in the creative activities. This is similar to the definition of mixed-initiative co-creativity (MI-CC) by Yannakis et al. (2014), who define it as the creation of artefacts with the interaction of a human and a computational initiative. They note that the two participants do not need to contribute to the same degree, and we do not demand symmetric contributions from human-computer co-creative systems neither.

The focus of this paper is on investigating the design processes for human computer co-creation systems. More specifically we investigate the transformation of machine creativity methods into co-creative ones, i.e., from batch methods to human-computer co-creation. Our goal is to shed light on the design process, key design decisions, and various issues in such transformation projects. We look at the process from two directions: a user-centered perspective and a computational creativity perspective based on Wiggins' (2006) model.

We first give a brief introduction to user-centered design and a brief description of Wiggins' model of computational creativity. We then carry out an investigation of three systems described in the literature. We discuss the observations, and then reflect our findings by comparing them to our ongoing work to produce interactive, educational poetry writing software for children.

User-Centered Design Perspective to Human-Computer Co-Creation

We are interested in methodologies and tools for supporting human-computer co-creation. The design of computer support for creativity has been studied both in the fields of interaction design (e.g. Carroll and Latulipe (2009)) and computational creativity (e.g. Yeap et al. (2010)). Interaction design and especially user-centered design can provide us with a well defined design process and a selection of documented methods, which have been demonstrated useful in designing real-life interactive software. Therefore we adopt user-centered design as the methodological framework for examining the work presented in this paper.

User-centered design (UCD) can be considered as "the active involvement of users for a clear understanding of user and task requirements, iterative design and evaluation, and

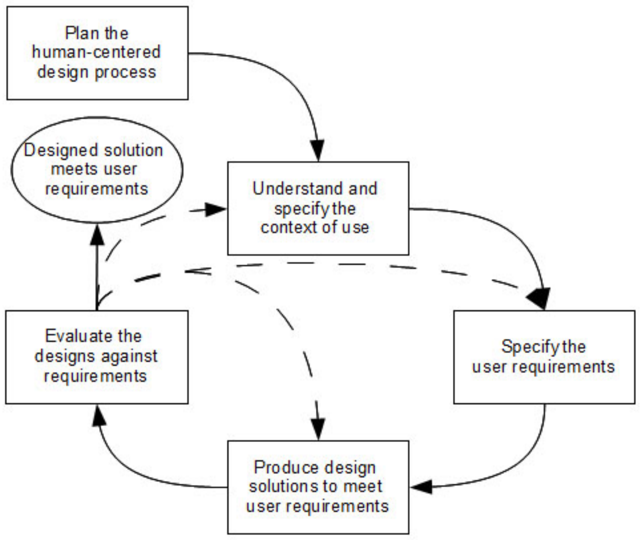
Plan the human-centered design process
Designed solution meets user requirements
Understand and specify the context of use
Evaluate the designs against requirements
Specify the user requirements
Produce design solutions to meet user requirements

Figure 1: The user-centered design process as specified in (ISO/IEC 2010)

a multi-disciplinary approach" (Vredenburg et al. 2002). UCD methods have been developed since the 1980s and are today "generally considered to have improved product usefulness and usability" (Vredenburg et al. 2002). UCD can also be viewed more broadly as a part of Interaction Design — an umberella term covering multiple disciplines emphasising different design perspectives in and outside of Human Computer Interaction (Rogers, Sharp, and Preece 2011, p. 9-11).

The UCD process (ISO/IEC 2010) contains six steps (Figure 1): (1) Plan the human-centered design process, (2) Understand and specify the context of use, (3) Specify the user requirements, (4) Produce design solutions to meet user requirements, (5) Evaluate designs against requirements, and (6) Designed solution meets user requirements. Steps 2 to 5 form an iterative circle in which step 5 can be followed again by steps 2, 3, or 4 until the requirements have been satisfied as presented.

Methods in UCD vary in level of user involvement, need of resources and type of gathered data as well as in which part of the design process they are most commonly utilised. Some of the methods are developed specifically by human-computer interaction specialists, and some are used by other human-oriented fields such as antrophology, as well. Usually each UCD team chooses methods suitable for the study of their users in the set context according to their own resources and expertise. The most used methods include iterative design, usability evaluation and informal expert review (Vredenburg et al. 2002). Many more exist and we encourage the interested reader to consult a handbook.

A Search Perspective to Creativity

From a computational creativity perspective, we can study creative behaviour supported by software in the light of Wiggins' formalization of creativity as search (Wiggins 2006).

Wiggins' model attempts to clarify and formalize some concepts in Margaret Boden's (1992) descripive hierarchy of creativity. This model represents creative systems with a septuple $\langle \mathcal{U}, \mathcal{L}, [\![.]\!], \langle\langle .,.,. \rangle\rangle, \mathcal{R}, \mathcal{T}, \mathcal{E} \rangle$. Here Universe $\mathcal{U}$ refers to an abstract set of all possible artefacts, for instance poems. $\mathcal{R}$ refers to a set of rules, expressed in the language $\mathcal{L}$, which defines a subset of the universe $\mathcal{U}$ i.e. the conceptual space of the creative system in question. Traversal function $\mathcal{T}$ defines how search in the universe is performed and the evaluation function $\mathcal{E}$ assigns a value for (some) elements of the universe. This formalization allows describing exploratory creativity as search (primarily) in the conceptual space defined by $\mathcal{R}$ via traversal funtion $\mathcal{T}$ and evaluation function $\mathcal{E}$, whereas transformational creativity may be achieved, e.g., by modifying the rules $\mathcal{R}$ defining the conceptual space.

Wiggins' model provides one way to look at the co-creative process between the user and the computer and to study interaction in the process. For instance, issues arising from conflicts between the rules, evaluation functions, and traversal functions of the computer and the user can now be clearly described in Wiggins' formalism. The (transformative) actions the user and the computer take when such conflicts appear decide what the rules, evaluation function, and traversal function of the larger system consisting of both the computer and the user are.

It has to be noted that many other theories, for instance the work by Csikszentmihalyi (1997), could be used as a viewpoint to look at co-creativity. However, we selected Wiggins' model for its rigorous nature and popularity in the field of computational creativity.

Case Studies

In this section we review three case studies of interactive software supporting human-computer co-creation. We first describe the criteria used for selecting these systems and then proceed to give a brief overview of the systems. We then analyse these three systems, in terms of design processes, user interactions and changes to the underlying machine creativity methods, which provides suggestions for developing future co-creative systems.

Since there are few descriptions of the design processes of human-computer co-creative systems in literature, we have used somewhat loose criteria to select software for this study:

1. The project utilises established methods of computational creativity.
2. The end result of the project is interactive with a human user.
3. Design decisions taken in the project are described.
4. Quantitative or qualitative feedback is available for the interactive software.

The above criteria emphasize projects drawing influences from both disciplines, computational creativity and human-computer interaction. Based on the criteria, we selected three systems: STANDUP (Ritchie et al. 2007; Waller et al. 2009), Scuddle (Carlson, Schiphorst, and Pasquier 2011), and Evolver (DiPaola et al. 2013). Our focus on the design process excludes some otherwise interesting examples of human-computer co-creative software, such as the Sentient Sketchbook (Liapis, Yannakakis, and Togelius 2013) and Tanagra (Smith, Whitehead, and Mateas 2011).

Overview of the selected systems STANDUP is a pun generating "language playground" developed for children with complex communication needs (CCN) (Ritchie et al. 2007; Waller et al. 2009). It is built on the basis of the JAPE system (Binsted 1996; Binsted and Ritchie 1997; 1994), which generates different classes of punning riddles using symbolic rules and a large, general purpose lexicon. The evaluation of the system with its target users suggested some restrictions in the capacity of the program but an increased facility with words and apparent enjoyment from its users (Waller et al. 2009). In addition, anecdotal evidence supported a positive effect on the communication of the children (Ritchie et al. 2007).

Scuddle is a movement exploration tool for choreographers to use in the early stages of their choreographic creation process (Carlson, Schiphorst, and Pasquier 2011). It is based on a genetic algorithm used to generate diverse combinations of movements. The evaluation of the program yelded positive results: users found the movements presented by the program non-habitual and creative and it prompted them to re-examine their own approaches to movement construction.

Evolver is a tool designed to help interior designers to explore design options based on the initial design elements provided by the designers themselves (DiPaola et al. 2013). Its focus is on helping the labor intensive early stages of a design project and offering novel designs outside the capabilities of its users. It is based on the autonomous creative genetic programming system called DarwinsGaze (DiPaola and Gabora 2009). Evolver was well received by its target audience who reported it supporting their creative processes, suggesting novel alternatives, easing manual work, and enabling communication. Interestingly some of the interior designers involved in the evaluation also considered the program as a collaborative partner in design instead of a mere platform.

All three systems show some established methods of computational creativity used as part of an interactive system. All systems have also been fairly successfull tools in increasing the creative potential of their users: STANDUP made the creative process of joke invention more accessible to an audience restricted by communication ability, Scuddle prompted new lines of creative inquiry in its users, and Evolver was at best considered a creative partner.

Interaction The level of user interaction is quite varied among the three cases. Of the three examples, Scuddle has the lowest level of interactivity. It provides the users only with simple options of starting or continuing the evolutionary algorithm, re-starting the whole process, or viewing six results evaluated by the computer (Carlson, Schiphorst, and Pasquier 2011). Describing these interaction options in Wiggins' framework, the theoretical categorisations of dance movements and their value can be seen as the conceptual space of the creative system. Traversal in the conceptual space is performed via a genetic algorithm which can be restarted or continued by the user evaluating the computer's pre-evaluated results. The user's role in the interaction lies more in the final evaluation of the artefacts than in the traversal of the options.

STANDUP has a higher level of interactivity than Scuddle. It offers a dual mode of interaction: user control can be divided into (1) options for the end user — a child with CCN, and (2) options for his or her carers. The child can choose a specific word to be included in the joke, a topic for the joke, or a specific joke type to be generated. The carer can adjust the program to suit the child best by restricting joke types, adjusting the words used in jokes based on their familiarity, or banning offensive words (Ritchie et al. 2007). In Wiggins' terms, the STANDUP user participates in defining the rules $\mathcal{R}$ in addition to participating in the transition function $\mathcal{T}$ and the evaluation function $\mathcal{E}$. On the other hand the computer provides the general conceptual space by defining the classes of puns and the allowed vocabulary. These can be modified by the user, i.e., the users' set of rules for conceptual space changes the respective set of rules of the computer. The traversal function of the computer is supervised by the user. The evaluation function of the computer makes sure that similar jokes have not been presented to the user before. The user makes the final evaluation and decides which of the jokes are saved.

Evolver provides the highest level of user interaction. The user provides the evolutionary algorithm with seed material and can select candidates to be used for generating the next generation of candidates as well as adjust the color scheme used (DiPaola et al. 2013). Viewed through Wiggins' framework, Evolver's interaction capabilities make the user's actions an integral part of the creative system: Evolver uses the seed material provided by the user to define the conceptual space. Traversal in this space is then performed via an evolutionary algorithm interactively with the user so that the user decides the parents for the next generation. The evaluation function of the co-creative system is a combination of the fitness function of the computer system and the final evaluation by the user.

Mapping the systems into Wiggins' model reveals that the human and the computer participating in the creative act can be viewed as one human-computer co-creative system. The mapping shows how both parties take responsibility over the generation of the creative artefact, although roles of the computer and the human are different. These particular examples also seem to indicate that the more interactive the system, the more integral the part of the user is in the creative model.

Design processes Carlson et al. (2011) started their design process for Scuddle by studying other computer aided choreographic systems and used the theory of choreography to establish requirements for Scuddle. They then proceeded to construct a prototype, which was tested with seven coreographers in simulated work sessions between a coreographer

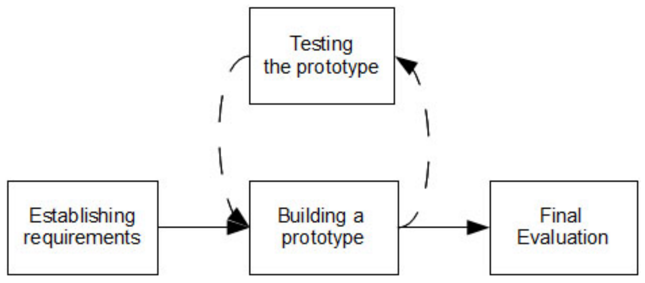

Figure 2: The design process of a co-creative tool described through the major design stages identified in the example projects

and a dancer. As evaluation methods they chose participant-observation and open ended interviews.

DiPaola et al. (2013) partnered with a design firm to develop Evolver. The design process started with establishing requirements by analysing the work processes of the employees of the partnering firm. The process continued with iterative prototyping and ended with a final evaluation conducted some months after the completion of the software.

Waller et al. (2009) relied on experts for gathering requirements for STANDUP. They continued iterative prototyping with the experts and adults with CCN and used typically developing children in testing graphics. The end product itself was evaluated with nine children with CCN during a ten week period including pre- and post-testing for the evaluation of learning effect, a training period for the children, and finally a scenario based observation of the users while using the software. The effects of the STANDUP software on the lives of the children beyond this period were studied with semistructured interviews and questionnaires directed at parents and other adults tightly involved with the children's learning progress.

All of the sample projects seem to follow a similar pattern in their design process (Figure 2). Each project starts by a requirement establishing stage and continues into prototype building. Two of the projects, Evolver and STANDUP continued this process iteratively by testing the prototype multiple times and adjusting it accordingly, while only one evaluation was conducted for Scuddle. The last iteration of this cycle can be called the final evaluation, a stage in which the final version of the prototype is evaluated more rigorosly, perhaps including assessment of usefullness or impact on the users.

When the process used in the studied cases is compared to the UCD process of Figure 1, we see that both processes share the stages of specifying requirements, producing solutions and evaluation. Both processes also have iterative properties, while the sample projects seem not to repeat the requirements setting stage. The stages of planning the process and understanding and specifying the context are missing from the case based description, but this may also be due to the result oriented reporting style of the papers, which may omit seemingly obvious details. Waller et al. (2009) report specifically having followed the UCD approach in designing STANDUP, and DiPaola et al. (2013) included researchers with a background in human-computer interaction in the design of Evolver.

Finally the processes differ in one important regard: If we categorise the processes by their starting points, Scuddle shows an example of applying a set of machine creativity methods directly into building interactive software, while Evolver and STANDUP both show an example of a process transforming existing autonomous creative systems into interactive products.

Changes to machine creativity methods To enable a higher level of interaction, the two projects using existing computational creativity prototypes had to conduct major changes in the machine creativity methods. These changes can be categorized into two rough categories: (1) changes done to facilitate interaction and (2) changes done to enhance the technical properties to better suit real-time use. The distinction between these classes can also be viewed through Wiggins' model. The first type of changes, driven by the goal of adding user interaction possibilities, increases the role of the user in Wiggins' model for the co-creative system, while the technical changes do not. However, the technical changes may support the quality of user interaction, which makes their categorisation without Wiggins' model difficult.

Ritchie et al. (2007) state that JAPE had multiple deficiencies which the STANDUP team had to account for by changing the system. The changes done to facilitate interaction in JAPE include keeping a record of jokes offered to a user to avoid too similar ones, the restriction of vocabulary to avoid obscene words and to focus on familiar ones, and possibilities to guide the search for jokes to a topic or specific words. The technical changes relate to adding better phonetic similarity measures and dropping some joke options to enhance the quality of jokes, as well as dropping some mechanisms to make the algorithm faster.

The DarwinsGaze algorithm underwent major changes in order to better suit the needs of Evolver's target audience as well (DiPaola et al. 2013). There is not as clear a distinction between interaction facilitating and technical changes on the surface, but viewed through Wiggins' model we see that giving the user control over the seed material and selection of candidates for pairing and adjusting the population both increase the user's role in the system. In addition, to emphasize gene linkage and user interpretability, the genetic algorithm was simplified by changing the gene structure to operate on a higher level of components called "design elements". The team also changed the internal format of pictures from bitmap to SVG to support layers in the generation and facilitate the import and export of pictures. Both of these modifications change the system in a way that can be seen in Wiggins' model. However, while the modifications increase the usability of the system, the user's role is not increased.

Building a Co-Creative Poetry Writer

We now move on to describe our on-going project developing an interactive poetry writing tool based on existing poetry generation software.

We chose children in comprehensive education as our target user group, as they are learning to use language in creative ways and explore much of the similar structures such as rhyme and rhythm, which are addressed by the existing creative software. The following sections examine our process and compare it to the example cases.

Basis in Computational Creativity Methods The machine creativity elements in the interactive system under construction are based on the poetry generation work by Toivanen et al. (2012). This approach uses corpus-based methods to find associated words around a given topic word and then to write poetry about the topic by using these words to substitute words in a given piece of text. Poetic devices like rhyming and alliteration can be further controlled by using constraint-programming methods (Toivanen, Järvisalo, and Toivonen 2013). In addition to these approaches, the system includes methods which can provide poetic fragments in a certain meter (e.g. iambic pentameter) and contain certain words. These fragments have been automatically extracted from large masses of text and different combinations of them, possibly modified with the word substitution method, can be used as a building block of poetry writing.

Design Process After choosing school children as the target audience, we started establishing requirements by studying the users and the context. Restricted by time and targeting a very sensitive group of users, we decided, like Waller et al. (2009), to rely on indirect input from children in our early design phases and use their participation only in the evaluation.

We recruited five enthusiastic grammar school teachers to help us. They kindly allowed us to observe their classes. Four of the teachers were teaching a group of approximately 70 second grade students together. One teacher specialised in the Finnish language and literature, teaching multiple classes in the 7-9th grade. We observed one full day of education in the second grade classroom, as well as two ninth grade lessons. We focused on observing interactions between the teachers and the pupils, as well as between pupils and how they worked on creative writing tasks on computers. After the lessons we conducted semi-structured interviews with the teachers in charge. We also sent an internet based open-ended questionnaire on teaching materials to the teachers.

The observation revealed differences in the skills of children: Younger children were generally still honing their skills in basic writing, whereas older children were more focused on the subject matter. The younger children were also challenged by foreign language user interfaces but quick to learn by trying things out and learning from their neighbours. The observation also showed in real contexts the behavior and language used by the children when communicating peer-to-peer or with the teachers. This experience gave us inspiration for selecting suitable interaction metaphors — connections to real world situations or objects, which help designing insightful interfaces — as well as for reducing the level of complexity in the user interface of our application. — as well as We expanded our observations with a literature study on educational software, which revealed more suitable interaction patterns and methods. The interviews and questionnaires showed that teachers saw technology as a means to motivate and aid the learner. Some teachers, especially those working with younger children and children with special education needs, expressed a need for quality software to aid the learning of writing. In general, teachers emphasised poetry writing's role as a creative activity.

The interviews and observation indicated that the writing skills of children develop highly individually. Therefore our software needs to cater for writers capable of different levels of creative writing. We decided to develop a creative writing tool allowing for a varied level of computer assistance, to enable writers with different skillsets to try out poetry writing. We decided to use fridge magnets as a simple metaphor for the manipulation of text on screen. An interface for writing sentences using the magnet metaphor has previously been successfully developed by Kuhn et al. (2009).

To test the design, we developed a paper prototype which we evaluated with a specialist researching the use of information technology in education. Based on her feedback we simplified the interface further and revised some features in saving and exporting poetry. She also noted that more advanced writers would need more abstract topics for writing than those we offered in our paper prototype. We iterated the paper prototype development until both the specialist and we ourselves were confident in building a working prototype.

At the moment of writing this, we are completing the prototype implementation. Next, we will evaluate the prototype in two ways: (1) scenario-based evaluations with pairs of children in a laboratory setting and (2) testing in a classroom. The former is designed to catch the troubles children might have with the tool and in the latter we want to see how teachers manage a learning setting using the software.

The early decisions made about methodology and user involvement can be interpreted as the planning phase of our project viewed through the UCD process. The observation, interviews, questionnaires and the literature study conclude the second and the third phase of the UCD process, or they can be interpreted as the first stage of the general process seen in the examples. The paper prototyping shows some of the iterative prototyping of the general process, or one iterative cycle of the UCD process returning from phase five to phase three. Finally the planned evaluation fits the general process lifted from the examples very well, while also following the lines of the UCD process.

However there are some challenges to following the UCD process to the letter: we found it challenging to communicate the restrictions of the computational approach to our users for ideation. Similarly, we found that it is difficult to create extensive paper prototypes for testing with users in iterative prototyping. This is mainly because the use cases by definition involve creative input from the user, and it is hard to imitate quick responses to creative inputs. This reduces the feasibility to include users in the early stages of design.

Interaction In a typical use case, the user can give the computer inspirational keywords, around which the computer generates a few lines of draft poetry which the user can then start to modify and extend. This should help the user past the "blank page" stage. The user may additionally ask for more lines, or just new words for a specific place in the poem to help find suitable rhyming pairs for example. Any new fragments of text produced by the system adapt automatically to the modifications and additions done by the user. To enable more symmetric human-computer co-creation, we are also experimenting with different ways to show editing suggestions to the user.

From the perspective of Wiggins' model, the user and the software share the same universe $\mathcal{U}$ and language $\mathcal{L}$, and they produce a poem together by editing it in turns. Traversal in the conceptual space can thus be performed both by the computer (e.g. providing a line of poetry or proposals for rhyming words) and by the user (e.g. adding more text or changing existing words). They both aim to satisfy (or modify and satisfy) their own rules $\mathcal{R}$ and evaluation function $\mathcal{E}$. This shows that our system can also be interpreted as a co-creative system with the user and the computer both sharing responsibility over the creative artefact.

Changes into the machine creativity methods The methods by Toivanen et al. (2012; 2013) were designed to compose poetry autonomously and certain changes were needed to modify them to work in an interactive system.

The interactive poetry writing process supports turns of word substitution and moving by the user and the computer. The grammar template needs to be updated when the user moves the words around.

The user may ask for suggestions for certain words and here the constraint-based methods need to be modified so that they can provide, for instance, suggestions for rhyming or alliterating words which satisfy some additional constraints like having a certain part-of-speech and grammatical case. Finally, the computer also needs to be able to update its vocabulary and keep record of the changes made by the user.

In Wiggins' terms, the rules $\mathcal{R}$ and the transition function $\mathcal{T}$ are defined in collaboration by the user and the computer as they both can change the contents of the poem. On the other hand, the evaluation is mainly done by the user as the computer evaluates only some things such as metric structure, and the final evaluation is always done by the user.

Conclusions

We have looked at the re-design of machine creativity methods into interactive human-computer co-creation tools. Based on the small sample of design processes that we studied, UCD methods seem to be common in creating interactive software on the basis of machine creativity methods. All of the cases we studied follow a similar process that can be viewed as an instance of the UCD process. However, the principles of user involvement, iterative design and a multidisciplinary approach are fulfilled to different extent in each project.

Computational creativity methods also set some boundaries for the software to be designed. However, as two of the case studies and our own project show, the methods can be re-negotiated for interactivity, transformationally changing the boundaries of interaction. This again can permit new designs making the re-negotiation process also iterative.

When characterised in Wiggins' framework, the observations are that for a high level of interactivity the re-negotiation of the methods must include interaction facilitating changes, which give the user a larger role in the system, and that only usability factors can be enhanced without expanding the role of the user in the re-negotiation. However, our sample is small and the search for other ways to increase interactivity demands further research.

The re-negotiation of computational creativity methods and the role of the user in them is an important part of defining the nature of creative interaction in the software. The design choices taken in the re-negotiation further define the extent to which we can achieve human-computer co-creation. These design choices may include questions such as whether the interactions are always human initiated, or if the computer may also spontaneously offer new creative perspectives, whether the interaction is done by exchanging creative artefacts, is instruction oriented, or is carried out in a more conversational manner creating a socio-technical environment resembling that of human-human co-creation.

UCD is focused on the human user. However, if we want to create more balanced human-computer co-creation, we may also need to account for the input the computer needs from the user to be able to participate in the process more extensively. Thus, it might be useful to look into collaborative creativity tools and remote presence to see if the computer can take a role similar to another human being as a creative collaborator. The roles of the user and the computer in co-creation should also be connected to the roles considered by Maher (2012).

Finally, interesting insight into human-computer co-creation could be gained by using Wiggins' framework to characterise interactions and their effects. Assume that the human and computer agents both apply their own traversal functions $\mathcal{T}$ on a shared (partial) artefact, based on their own rules $\mathcal{R}$ and evaluations $\mathcal{E}$. This can result, for instance, (1) in immediate synergy, such as reaching good areas in the search space that neither one can reach alone ("increasing generative inspiration"), (2) in pressure/possibility for transformational creativity (e.g. "productive aberration"), as well as (3) in conflicts where one agent takes the search into an area where the other one is not able to operate in a meaningful way ("generative uninspiration"). An analysis of such cases could provide guidance for issues that one should be able to deal with in human-computer co-creation.

Acknowledgments

This work has been supported by the Algorithmic Data Analysis (Algodan) Centre of Excellence of the Academy of Finland and the Helsinki Doctoral Program in Computer Science (HECSE).

References

Binsted, K., and Ritchie, G. 1994. An implemented model of punning riddles. In *AAAI*, 633–638.

Binsted, K., and Ritchie, G. 1997. Computational rules for generating punning riddles. *Humor* 10(1):25–76.

Binsted, K. 1996. *Machine humour: An implemented model of puns*. Ph.D. Dissertation, University of Edinburgh, Edinburgh, Scotland.

Boden, M. 1992. *The Creative Mind*. London: Abacus.

Carlson, K.; Schiphorst, T.; and Pasquier, P. 2011. Scuddle: Generating movement catalysts for computer-aided choreography. In *Proceedings of the Second International Conference on Computational Creativity*.

Carroll, E. A., and Latulipe, C. 2009. The creativity support index. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '09, 4009–4014. New York, NY, USA: ACM.

Csikszentmihalyi, M. 1997. Flow and the psychology of discovery and invention. *HarperPerennial, New York*.

DiPaola, S., and Gabora, L. 2009. Incorporating characteristics of human creativity into an evolutionary art algorithm. *Genetic Programming and Evolvable Machines* 10(2):97–110.

DiPaola, S.; McCaig, G.; Carlson, K.; Salevati, S.; and Sorenson, N. 2013. Adaptation of an autonomous creative evolutionary system for real-world design application based on creative cognition. In *Proceedings of the Fourth International Conference on Computational Creativity*, 40.

Fischer, G.; Giaccardi, E.; Eden, H.; Sugimoto, M.; and Ye, Y. 2005. Beyond binary choices: Integrating individual and social creativity. *International Journal of Human-Computer Studies* 63(4):482–512.

ISO/IEC. 2010. Iso 9241-210 ergonomics of human-system interaction – part 210: Human-centered design for interactive systems.

Kuhn, A.; Quintana, C.; and Soloway, E. 2009. Storytime: a new way for children to write. In *Proceedings of the 8th International Conference on Interaction Design and Children*, IDC '09, 218–221. New York, NY, USA: ACM.

Liapis, A.; Yannakakis, G.; and Togelius, J. 2013. Sentient sketchbook: Computer-aided game level authoring. In *Proceedings of ACM Conference on Foundations of Digital Games*.

Lubart, T. 2005. How can computers be partners in the creative process: classification and commentary on the special issue. *International Journal of Human-Computer Studies* 63(4):365–369.

Maher, M. L. 2012. Computational and collective creativity: Whos being creative?. In *Proc. 3rd Int. Conf. on Computational Creativity*.

Ritchie, G.; Manurung, R.; Pain, H.; Waller, A.; Black, R.; and OMara, D. 2007. A practical application of computational humour. In *Proceedings of the 4th International Joint Conference on Computational Creativity*, 91–98.

Rogers, Y.; Sharp, H.; and Preece, J. 2011. *Interaction Design: Beyond Human Computer Interaction*. Wiley, 3rd edition edition.

Smith, G.; Whitehead, J.; and Mateas, M. 2011. Tanagra: Reactive planning and constraint solving for mixed-initiative level design. *Computational Intelligence and AI in Games, IEEE Transactions on* 3(3):201–215.

Toivanen, J. M.; Toivonen, H.; Valitutti, A.; and Gross, O. 2012. Corpus-based generation of content and form in poetry. In *International Conference on Computational Creativity*, 175–179.

Toivanen, J. M.; Järvisalo, M.; and Toivonen, H. 2013. Harnessing constraint programming for poetry composition. In *International Conference on Computational Creativity*, 160–167.

Vredenburg, K.; Mao, J.-Y.; Smith, P. W.; and Carey, T. 2002. A survey of user-centered design practice. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, 471–478. New York, NY, USA: ACM.

Waller, A.; Black, R.; O'Mara, D. A.; Pain, H.; Ritchie, G.; and Manurung, R. 2009. Evaluating the standup pun generating software with children with cerebral palsy. *ACM Trans.Access.Comput.* 1(3):16:1–16:27.

Wiggins, G. A. 2006. Searching for computational creativity. *New Generation Computing* 24(3):209–222.

Yannakakis, G. N.; Liapis, A.; and Alexopoulos, C. 2014. Mixed-initiative co-creativity. In *Proceedings of the ACM Conference on Foundations of Digital Games*.

Yeap, W. K.; Opas, T.; and Mahyar, N. 2010. On two desiderata for creativity support tools. In *Proc. of the Intl. Conference on Computational Creativity*, 180–189.