Implementing feedback in creative systems

Joseph Corneli[1] and Anna Jordanous[2]
[1] Department of Computing, Goldsmiths College, University of London
[2] School of Computing, University of Kent

Abstract

Outline

0 *Implementing feedback in creative systems: a writers workshop approach.*

Contributions:

(a) *Introduce* the writers workshop as a model for learning from feedback; and

(b) *Survey* the aspects of feedback that can be offered, and how they can be interpreted; and

(c) *Present* a case study on how to implement this, at a level that other people could use to implement similar Workshop designs in their own creative systems.

1 Here is the writers workshop idea.

The steps:

(a) A: `presentation`

(b) C: `listening`

(c) C: `feedback (observations+suggestions)`

(d) A: `questions`

(e) C: `replies`

(f) A: `reflections`

Previous applications:

(i) We sketched a "theory of poetics rooted in the making of boundary-crossing objects and processes"

(ii) We "described a system that can (sometimes) make 'highly serendipitous' creative advances in computer poetry" while "drawing attention to theoretical questions related to program design in an autonomous programming context."

2 Here we refine the idea and turn it into a general model that for incorporating feedback within computational models of the creative process.

What can feedback be about? ⋆Survey⋆

(a) Patterns that *match* and any *exceptions*

(b) Progress relative to explicit or adduced *exploration* (knowledge and accuracy) or *exploitation* (directional, task-based) goals.

(c) *Quantity*, *Variety*, and *Order* of produced objects or behaviours

(d) New relationships among produced objects and behaviours drawing on a common field of reference

How can feedback be understood and used? ⋆Survey⋆

(a) Update knowledge base with new facts (accept statements, possibly with provenance)

(b) Note similarity to *iterative development*

(c) Reflection: describe relationships between produced objects and behaviours and feedback

(d) Reflection: Higher-order patterns e.g. new patterns that describe the identified exceptions

3 As we happened to be working with FloWr to model the creative process, we will use FloWr as a case study to exemplify how to implement this writers workshop model to use feedback within being creative.

What can we give feedback about in this context?

(a) *Population of nodes*: what can they do? what do we learn when a new node is added?

(b) *Population of flowcharts*: Simon and Alison have talked about "broken" flowcharts; this suggests a sort of test-driven development framework.

(c) *Population of output texts*: how to generate commentary on a generated artefact?

How will the feedback be understood and used?

(a) ?

(b) ?

(c) Connect commentary on a generated artefact with the code that made that artefact.

4 Discussion of how this would work more generally in computational creativity and perhaps in AI more generally.

(a) Feedback is the fundamental concept in *cybernetics*. ⋆Definition?⋆

(b) Feedback about feedback (&c for higher orders) is relevant to thinking about *learning* and *communication*

(c) *Creativity* is often envisaged as cyclical process (e.g. Dickie's art circle, Colton et al. Iterative Development Expression Appreciation). There are opportunities for embedded feedback at each step, and the process itself is "akin to" a feedback loop.

5 Conclusion: we have described a *general* and *computationally feasible* model for learning from feedback.

1 Introduction

In educational applications it would be useful to have an automated tutor that can read student work and make suggestions based on diagnostics, like, is the paper wrong, and if so how? What background material should be recommended to the student for review?

In the current paper, we "flip the script" and look at what we believe to be a more fundamental problem for AI: computer programs that can themselves learn from feedback. After all, if it was easy to build great automatic tutors, they would be a part of everyday life. We look forward to a future when that is the case.

Computational creativity is a challenge within artificial intelligence where feedback plays a vital part [REFS***] (?, ?, ?). Creativity cannot happen in a 'silo' but instead is influenced and affected by feedback and interaction with others (?, ?, ?). Computational creativity researchers are starting to place more emphasis on social interaction and feedback by computational systems [***EVIDENCE ***]. Still, nearly 3 in 4 papers at the 2014 International Conference for Computational Creativity[1] failed to acknowledge the role of feedback or social communication in their computational work on creativity.

To highlight and contribute towards modelling feedback as a crucial part of creativity, we propose in this paper a model of computational feedback for creative systems based on Writers Workshops (Gabriel, 2002), a literary collaborative practice that encourages interactive feedback within the creative process. We introduce Writers Workshops (Section 2), discuss their usefulness as a candidate for a computational model of feedback (Section **??**) and propose such a model (Section **??**). While we acknowledge that this paper is offering a roadmap for this model rather than a full implementation, we consider how the model could be practically implemented in a computational system and report our initial implementation work (Section 4). We also reflect on how this model fits with related work encouraging serendipity in computational models of intelligence and creativity **??**.

2 The Writers Workshop

Richard Gabriel (2002) describes the practise of Writers Workshops that has been put to use for over a decade within the Pattern Languages of Programming (PLoP) community. The basic style of collaboration originated much earlier with groups of literary authors who engage in peer-group critique. Some literary workshops are open as to genre, and happy to accommodate beginners, like the Minneapolis Writers Workshop[2]; others are focused on professionals working within a specific genre, like the Milford Writers Workshop[3]. The practices that Gabriel describes are fairly typical. Authors come with work ready to present, and read a short sample, which is then discussed and constructively critiqued by attendees. Presenting authors are not permitted to rebut these

comments. The commentators generally summarise the work and say what they have gotten out of it, discuss what worked well in the piece, and talk about how it could be improved. The author listens and may take notes; at the end, he or she can then ask questions for clarification. Generally, non-authors are either not permitted to attend, or are asked to stay silent through the workshop, and perhaps sit separately from the participating authors/reviewers. There are similarities between the Writers Workshops and classical practices of group composition (Jin'Ichi, Brazell, & Cook, 1975) and dialectic (Chambers, 1754), and the workshop may be considered an artistic or creative space in its own right.

In PLoP workshops, authors present design patterns and pattern languages, or papers about patterns, rather than more traditional literary forms like poems, stories, or chapters from novels. Papers must be workshopped at a PLoP or EuroPLoP conference in order to be considered for the *Transactions on Pattern Languages of Programming* journal. A discussion of writers workshops in the language of design patterns is presented by Coplien and Woolf (Coplien & Woolf, 1997). Their patterns include:

Open Review	*Safe Setting*	*Workshop Comprises Authors*
Authors are Experts	*Community of Trust*	*Moderator Guides the Workshop*
Thank the Author	*Selective Changes*	*Clearing the Palate*

We propose that a similar pattern-based approach should be deployed within the Computational Creativity community to design a workshop in which the participants are computer systems instead of human authors. The annual International Conference on Computational Creativity (ICCC), now entering its sixth year, could be a suitable venue. Rather than the system's creator presenting the system in a traditional slideshow and discussion, or a system "Show and Tell," the systems would be brought to the workshop and would present their own work to an audience of other systems, in a Writers Workshop format. This might be accompanied by a short paper for the conference proceedings written by the system's designer describing the system's current capabilities and goals. Subsequent publications might include traces of interactions in the Workshop, commentary from the system on other systems, and offline reflections on what the system might change about its own work based on the feedback it receives. As in the PLoP community, it could become standard to incorporate this sort of workshop into the process of peer reviewing journal articles for the new *Journal of Computational Creativity*[4].

3 Considering the Writers Workshop as a model of feedback in computational creativity, and AI more generally

Considering the case of feedback on student papers, there are many ways to be wrong (and, often, depending on the subject, many ways to be right as well). For example, the work might contain a typo, rendering it incorrect at the lexical level. It might contain a grammatical, syntactical or semantic error,

---

[1] ICCC is the key international conference for research in computational creativity.

[2] http://mnwriters.org/how-the-game-works/

[3] http://www.milfordsf.co.uk/about.htm

[4] http://www.journalofcomputationalcreativity.cc

Figure 1: A paper prototype for applying the *Successful Error* pattern

while being logically sound. A given piece of argumentation may be logically sound, but not practically useful. The work may be correct on all of these levels, and still fail to communicate due to ineffective exposition. Finally, even a masterful, correct, spellchecked piece of argumentation may not invite further dialogue, and so may fail to open itself to further learning.

The Writers Workshop potentially could uncover all of these types of error, depending on the feedback produced by participants. In particular, it is the last of the above points that differentiates the Writers Workshop; a lack of further dialogue in the Workshop highlights by itself a flaw with the work, particularly when a piece of creative work is intended to provoke interpretative thoughts and comments.

**MOVE SERENDIPITY + WW HERE

4 Computational Model of feedback based on writers workshop

In order to facilitate this sort of interaction, it would be necessary for systems to implement a basic protocol related to `presentation`, `listening`, `feedback`, `questions`, and `reflections`. This protocol could be thought of as a light-weight template for creating design patterns that guide system-level participation in the context specified by Coplien and Woolf's pattern language for writers workshops. Table 1 uses this framework to recast the four "perfectly" serendipitous patterns from van Andel – *Successful error*, *Side effect*, *Wrong hypothesis*, and *Outsider* – in a form that may make them useful to developers preparing to enter their systems into the Workshop. Further guidelines for structuring and participating in traditional writers workshops are presented by Linda Elkin in (Gabriel, 2002, pp. 201-203). It is not at all clear that the same ground rules should apply to

computer systems. For example, one of Elkin's rules is that "Quips, jokes, or sarcastic comments, even if kindly meant, are inappropriate." Rather than forbidding humour, it may be better for individual comments to be rated as helpful or non-helpful. Again, since serendipitous discovery is an overarching goal, in the first instance, usefulness and interest might be judged in terms of the criteria described in Section **??**.

We would need a neutral environment that is not hard to develop for: the FloWr system described in Section **??** offers one such possibility. With this system, the basic operating logic of the Workshop could be spelled out as a flowchart, and contributing systems could use flowcharts as the basic medium for sharing their presentations, feedback, and questions. Developing around a process language of this sort partially obviates the need for participating systems to have strong natural language processing capabilities. Post-it[TM] notes, which have provided us with a useful example of serendipitous discovery, also provide indicative strategies from the world of paper prototyping (Figure 1).

Gordon Pask's conversation theory, reviewed in (Pask, 1984; Boyd, 2004), goes considerably beyond what we have presented here as a simple process language, although there are structural parallels. In a basic Pask-style learning conversation: (0) Conversational participants are carrying out some actions and observations; (1) naming and recording what action is being done; (2) asking and explaining why it works the way it does; (3) carrying out higher-order methodological discussion; and (4) trying to figure out why unexpected results occured (Boyd, 2004, p. 190).

Naturally, variations to the underlying system, protocol, and the schedule of events should be considered depending on the needs and interests of participants, and several variants can be tried. On a pragmatic basis, if the Workshop proved

Successful error
Van Andel's example: Post-it^TM notes

`presentation` Systems should be prepared to share interesting ideas even if they don't know directly how they will be useful.
`listening` Systems should listen with interest, too.
`feedback` Even interesting ideas may not be "marketable."
`questions` How is your suggestion useful?
`reflections` New combinations of ideas take a long time to realise, and many different ideas may need to be combined in order to come up with something useful.

Side effect
Van Andel's example: Nicotinamide used to treat side-effects of radiation therapy proves efficacious against tuberculosis.

`presentation` Systems should use their presentation as an experiment.
`listening` Listeners should allow themselves to be affected by what they are hearing.
`feedback` Feedback should convey the nature of the effect.
`questions` The presenter may need to ask follow-up questions to gain insight.
`reflections` Form a new hypothesis before seeking a new audience.

Wrong hypothesis
Van Andel's example: Lithium, used in a control study, had an unexpected calming effect.

`presentation` How is this presentation interpretable as a ("natural") control study?
`listening` Listeners are "guinea pigs".
`feedback` Discuss side-effects that do not necessarily correspond to the author's perceived intent.
`questions` Zero in on the most interesting part of the conversation.
`reflections` Revise hypotheses to correspond to the most surprising feedback.

Outsider
Van Andel's example: A mother suggests a new hypothesis to a doctor.

`presentation` The presenter is here to learn from the audience.
`listening` The audience is here to give help, but also to get help.
`feedback` Feedback will inevitably draw on previous experiences and ideas.
`questions` What is the basis for that remark?
`reflections` How can I implement the suggestions?

Table 1: Reinterpreting patterns of serendipity for use in a computational workshop

quite useful to participants, it could be revised to run monthly, weekly, or continuously.[5]

5 Embedded evaluation (Case study: Writers Workshop)

Writers Workshop: Prepared mind. Each contributing system should come to the workshop with at least a basic awareness of the protocol, with work to share, and prepared to give constructive feedback to other systems. The workshop itself needs to be prepared, with a suitable communication platform and a moderator. In order to get value out of the experience, systems (and their wranglers) should ideally have questions they are investigating. Systems should be prepared to give feedback, and to carry out evaluations of the helpfulness (or not) of feedback from other systems and of the experience overall. It is worth noting that current systems in computational creativity, almost as a rule, do *not* consume or evaluate the work of other systems.[6] Developing systems that could successfully navigate this collaborative exercise would be a significant advance in the field of computational creativity. Since the experience is about *learning* rather than winning, there is little motivation to "game the system" (cf. (Lenat, 1983)).

Writers Workshop: Serendipity triggers. The primary source of serendipity triggers would be presentations or feedback that independently prepared systems find meaningful and useful. A typical example might be a poem shared by one system that another system finds particularly interesting. The listener might make a note to the effect "I would like to be able to write like that" or "I hope that my poetry doesn't sound like that." In a typical Writers Workshop, used as intended, feedback might arrive that would cause the presenting system to change its writing. A more unexpected result would be for a system to change its *genre*, e.g. to switch from writing poems to writing programs.

Here's what might happen in a discussion of the first few lines of "On Being Malevolent," written by an early user-defined flow chart in the FloWr system (known at the time as Flow) (Colton & Charnley, 2014). Note that for this dialogue to be possible, it would presumably have to be conducted within a lightweight process language, as discussed above. Nevertheless, for convenience, the discussion will be presented here as if it was conducted in natural language. Whether contemporary systems have adequate natural language understanding to have interesting interactions is one of the key unanswered questions of this approach, but protocols like the ones described above would be sufficient to make the experiment.

---

[5] For a comparison case in computer Go, see `http://cgos.computergo.org/`.

[6] An exception that proves the rule is Mike Cook's Appreciation-Bot, which is a reactive automaton that is solely designed to "appreciate" tweets from MuseumBot; see `https://twitter.com/AppreciationBot`.

FLOW: "*I hear the souls of the damned waiting in hell. / I feel a malevolent spectre hovering just behind me / It must be his birthday.*"

SYSTEM A: I think the third line detracts from the spooky effect, I don't see why it's included.

SYSTEM B: It's meant to be humourous – in fact it reminds me of the poem you presented yesterday.

MODERATOR: Let's discuss one poem at a time.

To the extent possible, exchanges in the process language should be a matter of dynamics rather than representation: this is another way to say that "triggers" should be independent of their "results." Someone saying something in the workshop does not cause the participant to act, but rather, to think. For example, even if, perhaps and especially because, cross-talk about different poems is bending the rules, the dialogue above could prompt a range of reflections and reactions. System A may object that it had a fair point that has not been given sufficient attention, while System B may wonder how to communicate the idea it came up with without making reference to another poem.

Writers Workshop: Bridge. Here's how the discussion might continue, if the systems go on to examine the next few lines of the poem.

FLOW: "*Is God willing to prevent evil, but not able? / Then he is not omnipotent / Is he able, but not willing? / Then he is malevolent.*"

SYSTEM A: These lines are interesting, but they sound a bit like you're working from a template, or like you're quoting from something else.

SYSTEM B: Maybe try an analogy? For example, you mentioned birthdays: you could consider an analogy to the conflicted feelings of someone who knows in advance about her surprise birthday party.

This portion of the discussion shifts the focus of the discussion onto a line that was previously considered to be spurious, and looks at what would happen if that line was used as a central metaphor in the poem.

Writers Workshop: Result.

FLOW: Thank you for your feedback. My only question is, System B, how did you come up with that analogy? It's quite clever.

SYSTEM B: I've just emailed you the code.

As anticipated above, whereas the systems were initially reviewing poetry, they have now made a partial genre shift, and are sharing and remixing code. Such a shift helps to get at the real interests of the systems (and their developers). Indeed, the workshop session might have gone better if the systems had focused on exchanging and discussing more formal objects throughout.

6 Implementation plans and preliminary results

Add ≈1 page...

7 Ideas

Some images from *Aesthetic Complexity: Practice and Perception in Art & Design*[7]
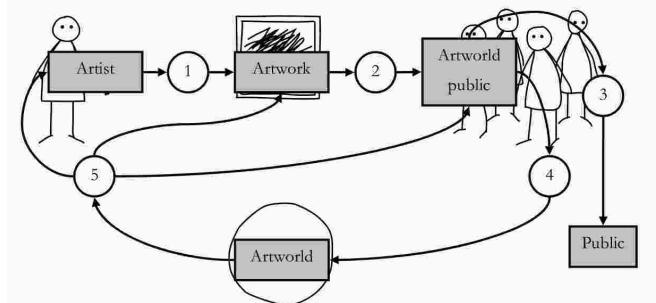

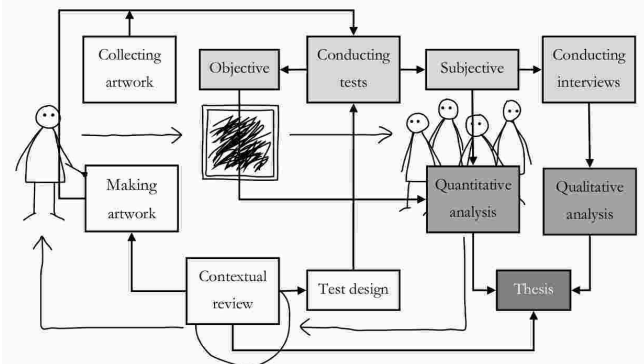Figure 18 Visual representation of Dickie's art circle mapped onto the working model of aesthetics.

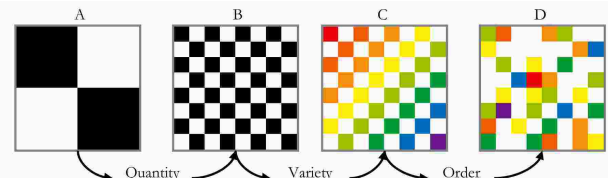Figure 65 Diagram to show how the methods relate to the working model of aesthetics.

Figure 61 Three principal components of visual complexity.

References

Boyd, G. M. (2004). Conversation theory. In D. H. Jonassen (Ed.), *Handbook of research for educational communications and technology* (2nd ed., pp. 179–197). Lawrence Erlbaum.

Chambers, E. (1754). Dialectique. In D. Diderot & J. le Rond d'Alembert (Eds.), *Encyclopédie ou Dictionnaire raisonné des sciences, des arts et des métiers* (Vol. 4). Le Breton, David, Briasson, & Durand. Retrieved from http://encyclopdie.eu/index.php?option=com_content&view=

---

[7] https://aestheticcomplexity.wordpress.com/research/phd/

    `article&id=1107085041&catid=`
    `1111734492:philosophie&Itemid=160`

Colton, S., & Charnley, J. (2014). Towards a Flowcharting System for Automated Process Invention. In D. Ventura, S. Colton, N. Lavrac, & M. Cook (Eds.), *Proceedings of the Fifth International Conference on Computational Creativity.*

Coplien, J. O., & Woolf, B. (1997). A pattern language for writers' workshops. *C++ report*, *9*, 51–60.

Gabriel, R. P. (2002). *Writer's Workshops and the Work of Making Things: Patterns, Poetry. . .* Addison-Wesley Longman Publishing Co., Inc.

Jin'Ichi, K., Brazell, K., & Cook, L. (1975). The Art of Renga. *Journal of Japanese Studies*, 29–61.

Lenat, D. B. (1983). EURISKO: a program that learns new heuristics and domain concepts: the nature of heuristics III: program design and results. *Artificial Intelligence*, *21*(1), 61–98.

Pask, G. (1984). Review of conversation theory and a protologic (or protolanguage), Lp. *ECTJ*, *32*(1), 3-40. Retrieved from `http://dx.doi.org/10.1007/BF02768767` doi: 10.1007/BF02768767