Salon des Refusés

Tomas Petricek
Alan Turing Institute, UK
tomas@tomasp.net

Stephen Kell
University of Cambridge, UK
stephen.kell@cl.cam.ac.uk

ABSTRACT

Most of academic work on programming languages is done in a way that makes it possible to evaluate the presented work using a small number of methods – an idea can be supported by a formal model with proofs, prototype implementation with measurable indicators or a controlled user study. As a result, programming language research is often shaped in a way that makes such evaluation possible. As a result, many interesting ideas about programming struggle to find space in the modern programming language research community, because we do not yet know how to evaluate them and we may even see them as "unscientific".

The workshop provides a venue where such ideas can be presented and discussed. In the absence of established evaluation methods, our only resort is to subject work to constructive critical review. This workshop takes inspiration from literary criticism – submissions that provoke an interesting discussion will be published together with an attributed review that presents an alternative position, develops additional context or summarizes discussion from the workshop.

The methodology of constructive critical commentary makes it possible to explore a wider space of programming ideas than those that are covered by established programming language conferences. This workshop not only enables exploration of new areas of the programming language ideas space, but also provides a venue for discovering other areas of the idea space where further quantitative or qualitative evaluation methods can be applied.

1. MOTIVATION

The scope of the RIOT workshop is delimited more by the methodology and style of submissions than by a particular sub-field of programming research. For this reason, the objectives below focus on the kind of work we encourage rather than on specific topics. The technical topics of the workshop will be party determined by the selection of the program committee (and can be found in the Call for Papers).

As mentioned above, we welcome ideas that are difficult to evaluate and might even be seen as "unscientific", but the value of exploring such ideas is supported by an overwhelming evidence from history and philosophy of science (discussed throughout the proposal).

1.1 Workshop objectives

The objective of the workshop is to explicitly provide venue for discussing programming language ideas that are difficult to evaluate using established evaluation methods. The kind of submissions we encourage and welcome are listed below.

THOUGHT EXPERIMENTS. Just like Galileo's early efforts involved thought experiments, analogies and illustrative metaphors rather than detailed experimentation[1], we believe that thought experiments can provide novel insights and inspire fruitful programming language ideas.

Wadler's widely cited "expression problem"[2] (which was never formally published) can be seen as such programming language thought experiment. It defines a context for assessing abstraction capabilities of type systems, but it does so without requiring concrete definition of what a type is[3].

EXPERIMENTATION. As noted by Hacking, we find prejudices in favour of theory, as far back as there is institutionalized science[4], but programming can often be seen more as experimentation than as theorizing.

The physicist George Darwin used to say that every once in a while, one should do a completely crazy experiment, like blowing the trumpet to tulips every morning for a month (or porting an application from Haskell to COBOL?). Probably nothing notable will happen, but if something did happen (the COBOL application becomes easier to use and has fewer bugs!), that would be a stupendous discovery[5]. We encourage submissions that report such stupendous discoveries, even if there is yet no overreaching theory that explains why and how they happened.

PARADIGMS. All scientific work is rooted in a scientific paradigm or scientific research programme[6]. Those define not only appropriate methods for answering scientific questions, but also determine what questions are asked. For example, the Algol research programme seeks to increase confidence in correctness by the use of formal methods[7].

We would like to encourage submissions that explore alternative scientific paradigms or research programmes by acknowledging that logically perfect versions of theories and results usually arrive long after imperfect versions have enriched science[8].

[1] Chalmers (1999), p106
[2] Wadler (1998)
[3] An extended discussion can be found in Petricek (2015)
[4] Hacking (1983), p150
[5] The original citation appears in Hacking (1983), p151
[6] Kuhn (1970) and Lakatos (1975), respectively
[7] Priestley (2012), p257
[8] Feyerabend (2010), p8

METAPHORS, MYTHS AND ANALOGIES. Any description of formal, mathematical, quantitative or even poetical nature still represents just an analogy[9], and despite the dominance of mathematical and quantitative analogies, we believe that there are fruitful ideas that can be learned from other forms of analogies[10]. After all, John von Neumann's First report on EDVAC[11], which introduced modern computer architecture, was inspired by a biological metaphor and referred to individual computer components as "organs".

FROM JOKES TO SCIENCE FICTION. Ideas first presented as science fiction stories or said as a joke may enrich serious science in unexpected ways. The idea that a steam engine could be used to execute laborious computations was first suggested "in a manner which certainly at the time was not altogether serious" sparking "serious consideration of the possibility of mechanical computation."[12]

A story or an artistic performance may explore ideas and spark conversations that provide crucial inspiration for development of new computer science thinking.

CONCLUSIONS. The format of constructive criticism (as discussed in the workshop format section) can provide a way for making discussions triggered by submissions covering the above (perhaps unorthodox) topics a valuable contribution to programming language research literature.

A secondary objective of the workshop is to begin a systematic exploration of the ideas in the programming language space. One of the roles of the critical reviews (published together with accepted submissions) is to provide additional context and relate the discussed ideas with other areas of the programming language design space.

1.2 Audience: Academics and programmers alike

The workshop provides a venue where unorthodox programming ideas can be discussed and we would like to attract diverse and open-minded audience with both academic and practical background.

- By accepting submissions that do not require established academic forms of evaluation, we want to create a venue that is welcoming to not just to novel ideas from the academic community, but also to practitioners.
- We welcome work that presents alternative perspectives on programming including, for example, treating spreadsheets as programming languages or live coded music (and treating other instruments as programming tools).
- We would like to attract submissions and attendees who are interested not just in programming, but also in philosophy of programming and we encourage submissions that are reflections on academic programming research.

In summary, we are convinced that opening the workshop to ideas that do not fit established programming language conferences and established evaluation methods can contribute to the diversity of ideas presented at ‹Programming›.

1.3 Relevance: Creating new open-minded venue

Similarly to the main ‹Programming› conference track, we welcome submissions covering a range of topics related to programming and we intend to contribute to the open-minded and innovative nature of the conference.

The Salon des Refusés workshop complements the main conference track by explicitly seeking submissions that do not present clear evaluation. We believe such submissions can present interesting and valuable ideas that would not be accepted at the main conference track, which requires strong evidence or compelling arguments.

We would like to provide an additional venue where work that expands our way of thinking about programming can be presented, provided that it sparks the interest of the program committee or other workshop attendees.

1.4 Context: Alternative programming venues

The workshop follows a number of informal discussions among the organizers and prospective program committee members that have taken place at a number of programming language-related conferences including SPLASH, Onward! (2015) and PPIG (2016) in which we discussed the restrictions that standard evaluation criteria place on programming language research (and the effect this has on what kind of programming research is undertaken by our community).

We believe that taking inspiration from literary criticism, soliciting papers that provoke interesting discussion and publishing them together with critical review written by other workshop attendees or program committee members (see evaluation process below) provides a way to take the ideas forward within a more structured academic format.

The workshop is related to a number of other events in the programming language community:

- Off the Beaten Track (OBT)[13] explores unconventional ideas in programming, however, it presents them more as "fun" side-projects (often using traditional programming theory methods in non-traditional context) rather than serious academic endeavour. Our workshop aims to find new ways of doing serious academic work.
- PPIG[14] and HaPoC[15] are two examples of conferences that explore aspects of programming rarely present at main-stream programming language venues (psychology, history and philosophy). Those cater well for specific programming-related topics, but do not cover a full spectrum of interesting programming ideas.

[9] von Foerster (2013)
[10] It is worth noting that we are not the only ones calling for exploration of other analogies in a heavily mathematicised science. For example, Sedlacek (2011) traces many economical concepts in ancient myths and argues for their relevance to modern economics.
[11] von Neumann (1945)
[12] Priestley (2011), p22
[13] http://conf.researchr.org/home/OBT-2016
[14] http://www.ppig.org/
[15] http://hapoc.org/

1.5 Need: Searching for evaluation methods

The workshop addresses the need for more flexible evaluation methods that would make it possible for programming language research to explore a wider range of ideas. One comment illustrating the need has been made recently by Sean McDirmid:

> *The design space for [programming languages] is so incredibly broad that we have just explored only a very tiny bit of it. But we seem to be stuck in those limited spaces we have explored since (…).*[16]

This workshop proposal presents an attempt at creating venue that would explicitly welcome such explorations of yet underexplored space of programming language ideas.

Such explorations are difficult to perform, because predominant forms of evaluation (which are directly or indirectly required when presenting results) direct the work back towards the already explored areas. As noted by Feyerabend:

> *To 'clarify' the terms of a discussion does not mean to study the additional (…) properties of the domain in question (…), it means to fill them with existing notions from the entirely different domain of logic (...). So the course of an investigation is deflected into the narrow channels of things already understood (…).*[17]

Encouraging alternative forms of submissions that are not restricted to "existing notions" of evaluation can provide the much needed space for exploring novel ideas in programming, including those that do not fit traditional evaluation methods:

> *If a subject does not permit exactness, it is not sufficient to be exact about something else*[18].

The alternative that we are seeking to establish is to restore to use once more the power for the deliberate holding of unproved beliefs[19], and provide space where the consequences of such unproved beliefs can be explored and taken further. We believe this approach is necessary to find new interesting points in the programming design space, other than those that are attracted to the currently accepted programming language research methodologies.

2 ORGANIZATION

WORKSHOP ORGANIZERS.

- Tomas Petricek (tomas@tomasp.net, primary contact), Alan Turing Institute, UK. Tomas recently submitted his PhD thesis at University of Cambridge and is currently working as a researcher at the Alan Turing Institute in London. Tomas is active in the industrial programming community and helped to organize a number of events in the industry (including a number of conferences focusing on the F# programming language[20] and also a virtual fsharpConf conference[21]).
- Stephen Kell (stephen.kell@cl.cam.ac.uk), University of Cambridge. Stephen is an active member of the programming language community and has experience with serving as a PC member at numerous workshops. He also served as a member for SPLASH workshops organizing committee.

PROGRAM COMMITTEE. The preliminary proposal has been discussed with a number of people who agreed to join us as program committee members:

- Dominic Orchard, University of Kent
- Felienne Hermans, TU Delft
- Antranig Basman, Raising the Floor – International
- Sam Aaron, University of Cambridge
- Luke Church, Google and University of Cambridge

We are in process of contacting other potential program committee members and we expect to finalize the list before publishing the workshop web site on November 1.

WORKSHOP FORMAT.

- We expect to follow the deadlines recommended in the call for workshops (with a submission deadline at the end of January and author notification on February 17).
- We intend to accept full papers, short papers and (possibly) other kinds of submissions (either screencasts or interactive web essays). For papers, we plan to use the official ‹Programming› paper template with the same page limit (22 pages) for full papers and 10 pages for short papers.
- We do not expect to have printed proceedings.
- For accepted papers, we plan to have 25-minute talk by the author, followed by a 20-minute review talk and further discussion led by the primary reviewer of the paper. This would let us have 6 accepted talks in a day together with (yet to be determined) keynote.
- As mentioned in the abstract, the evaluation process for the workshop is inspired by the format of literary criticism and is discussed in more detail below.
- We expect that around 10 people from our existing community would attend the workshop and we consider the workshop a success if it attracted further 20 attendees of the main conference, making the total number around 30.
- We will require data projector for the workshop, but no other special equipment.

[16] Sean McDirmid, retrieved 11 Oct 2016 from: http://lambda-the-ultimate.org/node/5140#comment-85268
[17] Feyerabend (2010)
[18] Perry (1954) as quoted by Polanyi (1958)
[19] Paraphrasing Polanyi (1958)
[20] http://fsharpworks.com/paris/2014.html and http://fsharpworks.com/mvp-summit/2015.html
[21] http://fsharpconf.com/

EVALUATION PROCESS. We intend to create venue that can provide place for presenting interesting and novel programming ideas that are difficult to evaluate using established evaluation criteria. For this reason, the evaluation process for workshop papers will focus more on how well the presented work explores (or inspires exploration of) novel points in the programming language design space. We see this as a valuable contribution on its own.

In the program committee, we intend to follow process inspired by "identify a champion"[22] where program committee members select papers they find interesting or worth discussing and exposing to constructive criticism. The champions will be acknowledged publicly and will be responsible for the second talk during the session dedicated to the paper at the workshop and also for writing critical review or commentary that will be published, together with the original work in the workshop proceedings.

This way of evaluating papers follows the core of scientific practice in that peer review is the key part of the process, but it accepts the fact (discussed by Polanyi[23]) that the value of scientific work cannot always be precisely articulated and often relies on personal commitment of an individual.

REFERENCES

Feyerabend, P. (2010). Against method. Verso (4th edition). ISBN 1844674428.

Hacking, I. (1983). Representing and Intervening: Introductory Topics in the Philosophy of Natural Science. Cambridge University Press. ISBN 0521282462.

Chalmers, A. F. (1999). What is this thing called science? Open University Press.

Kuhn, T. S. (1970). The Structure of Scientific Revolutions. The University of Chicago Press (2nd edition). ISBN 0226458040.

Lakatos, I. (1975). Falsification and the Methodology of Scientific Research Programmes. In Can Theories be Refuted? Essays on the Duhem-Quine Thesis (ed. Harding, S. G.), pp205-259. ISBN 9789027706300.

von Neumann, J. (1945). First Draft of a Report on the EDVAC. University of Pennsylvania.

Petricek, T. (2015). Against a universal definition of 'type'. In proceedings of Onward! Essays, ACM

Perry, R. B. (1955). Realms of Value: a Critique of Human Civilization.

Priestley, M. (2011). A science of operations: machines, logic and the invention of programming. Springer Science & Business Media.

Polanyi, M. (1958). Personal knowledge: Towards a post-critical philosophy. University of Chicago Press.

Wadler, P. (1998). The expression problem. Sent to the Java-genericity mailing list.

von Foerster, H., Pörksen, B. (2013). Pravda je výmysl lháře, Pragma, 9788073494315 (translated from Wahrheit ist die Erfindung eines Lügners: Gespräche für Skeptiker).

Sedlacek, T. (2011). Economics of good and evil: the quest for economic meaning from Gilgamesh to Wall Street. Oxford University Press. ISBN 9780199767205.

Nierstrasz, O. (2000). Identify the champion. Pattern Languages of Program Design 4 (2000): 539-556.

[22] Nierstrasz (2000)

[23] Polanyi (1958)

DRAFT CALL FOR PAPERS

TBD