Salon des Refusés
Dialectics for new computer science

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

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

Abstract. Most academic work on programming languages follows methodology that makes it possible to evaluate the presented work via a small variety of means – an idea can be supported by a formal model with proofs, a prototype implementation with measurable indicators, a case study, a controlled user study or interviews. Programming language research is incentivised to make such evaluation possible. As a result, many interesting ideas about programming might 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 – an idea brought to the programming research context at a recent PPIG workshop[1] – 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 that which is covered by established programming language conferences. This workshop enables such exploration of new areas and also serves as a concrete opportunity to explore new evaluation methods.

1. MOTIVATION

Salon des Refusés ("exhibition of rejects") was an 1863 exhibition of artworks rejected from the official Paris Salon. The jury of Paris Salon required near-photographic realism and classified works according to a strict genre hierarchy. Paintings by many, later famous, modernists such as Camille Pissarro and Édouard Manet were rejected and appeared in what became known as the Salon des Refusés[2].

A ‹Programming› workshop equivalent of Salon des Refusés can provide space for exploring new ideas and new ways of doing computer science. The scope of the workshop is delimited more by the methodology and style of submissions than by a particular sub-field of programming research. For this reason, the next section focuses on the kind of work we encourage rather than on specific topics. The technical topics of the workshop will be partly determined by the selection of the program committee (and can be found in the Call for Papers).

We welcome ideas that are difficult to evaluate and might even be seen as "unscientific". The value of exploring such ideas is supported by significant evidence from history and philosophy of science (discussed throughout).

1.1 Workshop objective

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

THOUGHT EXPERIMENTS. Just as Galileo's early efforts involved thought experiments, analogies and illustrative metaphors rather than detailed experimentation[3], we believe that thought experiments can provide novel insights and inspire fruitful programming language ideas.

Wadler's widely cited (and never formally published) "expression problem"[4] can be seen as such a 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[5].

EXPERIMENTATION. As noted by Hacking, we find prejudices in favour of theory, as far back as there is institutionalized science[6], 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 (it turns out that COBOL is not that bad after all!), that would

[1] Church (ed.). (2016)
[2] This brief overview is based on Wikipedia. We apologize to art-historians for any misinterpretations that we might have introduced. (http://en.wikipedia.org/wiki/Salon_des_Refuses, retrieved 21 Oct 2016)
[3] Chalmers (1999), p106
[4] Wadler (1998)
[5] An extended discussion can be found in Petricek (2015)
[6] Hacking (1983), p150

be a stupendous discovery[7]. We encourage submissions that report such stupendous discoveries, even if there is yet no overarching theory that explains why they happened.

PARADIGMS. All scientific work is rooted in a scientific paradigm or scientific research programme[8]. Those define not only appropriate methods for answering scientific questions, but also frame what questions can be asked. For example, the Algol research programme seeks to increase confidence in correctness by the use of formal methods[9].

We encourage submissions that explore alternative scientific paradigms or research programmes by acknowledging that logically perfect versions of theories usually arrive long after imperfect versions have enriched science[10].

METAPHORS, MYTHS AND ANALOGIES. Any description of formal, mathematical, quantitative or even poetical nature still represents just an analogy[11], and despite the dominance of mathematical and quantitative analogies, we believe that there are fruitful ideas that can be learned from other forms of analogy[12]. After all, John von Neumann's First report on EDVAC[13], 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."[14]

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

How can submissions covering the above unorthodox topics become a valuable contribution to programming research literature? The constructive criticism format of the workshop provides one such opportunity, by presenting unorthodox ideas in wider critical context (see workshop format below).

A secondary objective of the workshop is to promote determined exploration of the ideas in the programming language space. Critical reviews (published together with accepted submissions) provide additional context and relate the work with other ideas such as end-user design, programming systems and alternative idioms for computation.

1.2 Audience: Academics and programmers alike

The workshop provides a venue where unorthodox programming ideas can be discussed and we aim to attract diverse and open-minded audience with a range of backgrounds.

- By accepting submissions that do not require established academic forms of evaluation, we 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, e.g., treating spreadsheets as programming languages, live coded music and work that widens access to the affordances of programming.
- 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 traditional evaluations. 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 provide a venue where novel 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) which were centred around the restrictions that standard evaluation criteria place on programming language research (and the effect this has on what kind of research is undertaken by the 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.

[7] The original citation appears in Hacking (1983), p151
[8] Kuhn (1970) and Lakatos (1975), respectively
[9] Priestley (2012), p257
[10] Feyerabend (2010), p8
[11] von Foerster (2013)
[12] 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 concepts from economics in ancient myths and argues for their relevance to modern economics.
[13] von Neumann (1945)
[14] Priestley (2011), p22

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

- Off the Beaten Track (OBT)[15] 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 that accommodates both theoretical and practical knowledge.
- PPIG[16] and HaPoC[17] 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[18].

1.5 Need: Opening space for new ideas

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 (…).*[19]

This workshop proposal presents an attempt at creating a 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 (…).*[20]

Encouraging alternative forms of submissions that are not restricted to "existing notions" of evaluation can provide the much-needed space for exploring novel ideas, 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*[21].

The alternative that we aim to establish is to restore the power for the deliberate holding of unproven beliefs[22], and provide space where the consequences of such unproven 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 visiting researcher at the Alan Turing Institute in London. Tomas is active in the industrial programming community. He organized or helped to organize a number of events in the industry (including several conferences focusing on the F# programming language[23] and also a virtual fsharpConf conference[24]).
- Stephen Kell (stephen.kell@cl.cam.ac.uk), University of Cambridge. Stephen is an active member of the PL community and has experience with serving as a PC member at numerous events. He also served as a member for SPLASH workshops programme committee.
- In addition to the two main organizers, many of the prospective program committee members are involved in the core organization efforts of the workshop and will be able to help with the necessary organizational work. This includes Dominic Orchard (organizer of TVCS'16[25], program chair of PLACES'16 and co-chair of PLE'15 and PLE'14) and Felienne Hermans (organizer of Joy of Coding[26] which is a conference bridging the gap between academia and programming research).

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

[15] http://conf.researchr.org/home/OBT-2016
[16] http://www.ppig.org/
[17] http://hapoc.org/
[18] Evidence suggesting that PPIG does not cover the full spectrum of ideas we are aiming for can be found in a recent empirical review by the organizers of PPIG 2016 in Church et al. (2016)
[19] Sean McDirmid, retrieved 11 Oct 2016 from: http://lambda-the-ultimate.org/node/5140#comment-85268
[20] Feyerabend (2010)
[21] Perry (1954) as quoted by Polanyi (1958)
[22] Paraphrasing Polanyi (1958)
[23] For example: http://fsharpworks.com/paris/2014.html and http://fsharpworks.com/mvp-summit/2015.html
[24] http://fsharpconf.com/
[25] http://www.cl.cam.ac.uk/~dao29/meeting-tvcs/
[26] http://joyofcoding.org/

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 welcome short papers (up to 3000 words) and long papers (up to 9000 words) as well as screencasts or interactive essays). We intend to publish accepted paper on the web, but any format is welcome for the submission (authors can use the ‹Programming› paper template).
- All accepted papers will be published online on a dedicated workshop web page (using a web-browser friendly format). We do not expect to have printed proceedings.
- For accepted papers, we plan to have 20-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 up to 8 accepted talks in a day together with (yet to be determined) keynote.
- The evaluation process for the workshop is inspired by the format of literary criticism and is discussed in more detail in the following paragraph.
- 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 a data projector for the workshop, but no other special equipment.

EVALUATION PROCESS. We intend to create a venue that can give a place to 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 a process inspired by “identify a champion”[27] where program committee members select papers they find interesting or worth discussing and expose them to constructive criticism. The committee members championing the accepted papers will be acknowledged publicly and will be responsible for giving a second (critical) 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.

These means of engaging with papers follows the core of scientific practice in that peer review is the key part of the process, but it accepts the fact (see Polanyi[28]) 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.

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

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.

Church, L., Marasoiu, M. (2016). A fox not a hedgehog: What does PPIG know? In Proceedings of PPIG 2016.

Church, L. (ed.), (2016). Pre-print of the 27th Annual Workshop of the Psychology of Programming Interest Group. Available at http://www.ppig.org.

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.

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

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.

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

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

[27] Nierstrasz (2000)

[28] Polanyi (1958)

Call for Papers: Salon des Refusés – Dialectics for new computer science

Salon des Refusés ("exhibition of rejects") was an 1863 exhibition of artworks rejected from the official Paris Salon. The jury of Paris Salon required near-photographic realism and classified works according to a strict genre hierarchy. Paintings by many, later famous, modernists such as Édouard Manet were rejected and appeared in what became known as the Salon des Refusés. This workshop is the programming language research equivalent of Salon des Refusés. We provide venue for exploring new ideas and new ways of doing computer science.

Many interesting ideas about programming might struggle to find space in the modern programming language research community, often because they are difficult to evaluate using established evaluation methods (be it proofs, measurements or controlled user studies). As a result, they are often seen as "unscientific". Rather than requiring detailed evaluation, this workshop provides a venue where interesting and thought provoking ideas can be exposed to critical evaluation. Submissions that provoke interesting discussion among the program committee members will be published together with an attributed review that presents an alternative position, develops additional context or summarizes discussion from the workshop. This means of engaging with papers enables explorations of novel programming ideas and new ways of doing computer science.

Topics of interest

The scope of the workshop is determined more by the format of submissions than by the specific area of programming language or computer science research that we are interested in. We welcome submissions in a format that makes it possible to think about programming in a new way, including, but not limited to:

- Thought experiments – we believe that thought experiments, analogies and illustrative metaphors can provide novel insights and inspire fruitful programming language ideas.
- Experimentation – we find prejudices in favour of theory, as far back as there is institutionalized science, but programming can often be seen more as experimentation than as theorizing. We welcome interesting experiments even if there is yet no overarching theory that explains why they happened.
- Paradigms – all scientific work is rooted in a scientific paradigm that frame what questions can be asked. We encourage submissions that reflect on existing paradigms or explore alternative scientific paradigms.
- Metaphors, myths and analogies – any description of formal, mathematical, quantitative or even poetical nature still represents just an analogy. We believe that fruitful ideas can be learned from less common forms of analogies as well as from the predominant, formal and mathematical ones.
- From jokes to science fiction – a story or an artistic performance may explore ideas and spark conversations that provide crucial inspiration for development of new computer science thinking.

Format and review

We welcome short papers (up to 3000 words) and long papers (up to 9000 words) as well as screencasts or interactive essays. We intend to publish accepted paper on the web, but any format is welcome for the submission (authors can use the ‹Programming› paper template).

Key dates

- Deadline for submissions: February 1st 2017
- Notification of authors: February 17th 2017
- Early registration deadline: March 6th 2017
- Workshop at ‹Programming› 2017: April 3rd – 6th 2017

Program committee

Submissions that spark interesting discussion will be preferred over submissions with detailed evaluation. We asked the PC members to write about their interests in order to help authors find topics of positions that will be of interest to the PC.

DOMINIC ORCHARD (UNIVERSITY OF KENT). Dominic frequently works within the research paradigm of using mathematics and logic as tools for understanding programs and computation. He is fascinated by times when this activity feels like shoving a square peg in a round hole, presenting an opportunity to think outside, or against, the paradigm or seek better tools within it.

FELIENNE HERMANS (DELFT UNIVERSITY OF TECHNOLOGY). Felienne likes to think about what is and what is not programming. She especially loves to help people be better at programming, while they might not be actively looking to get better, because they do not self-identify as programmers. As such she has worked on code smells and refactoring for Excel and for Scratch, a programming language for children.

ANTRANIG BASMAN (RAISING THE FLOOR – INTERNATIONAL). Antranig wants to see work that widens the audience for software by considering the role it might take in healthy societies, based around artefacts that work for everyone. These days, he codes exclusively in JavaScript, the language of the proles - in whom our hope lies. He is interested in work which challenges the assumptions we use to carve up our domain into separated disciplines. He is excited by the possibility that we are still in the prehistory of our subject, and that the principles and practices we have adopted so far may be completely faulty.

STEPHEN KELL (UNIVERSITY OF CAMBRIDGE). Stephen thinks that programming, as we know it, has unacceptably high human cost, and that we cannot solve this problem by escalation. We need programming systems that help us not to write more code, but to write less, combine, downsize and simplify code. He is a system-builder, interested not only in designing and building such programming systems, but in evolving existing systems in this non-traditional direction.

SAM AARON (UNIVERSITY OF CAMBRIDGE). Sam is a live coder working directly at the intersections of art, education and programming language research. He is particularly interested in exploring the notion of liveness within languages enabling him to consider code as an interface for direct manipulation. He is the creator of Sonic Pi - a live coding music synthesiser currently gaining traction by both school teachers and musicians alike.

TOMAS PETRICEK (ALAN TURING INSTITUTE, LONDON). Tomas is interested in work that challenges how we think about programming. He is interested in novel programming models, theory and practice of functional programming, tools for data-driven storytelling and data science, but also philosophy of science applied to programming.

LUKE CHURCH (GOOGLE AND UNIVERSITY OF CAMBRIDGE). Luke Church is a researcher at Google and the University of Cambridge. He studies how to improve the experience that people have when dealing with complex systems. For example: programming languages, configuration systems or animal behaviour.