SUBSTRATES '25 // VISION STATEMENT

Author: Patrick Dubroy (https://dubroy.com)

What is a substrate?

This feels like an important question to answer!

To me, the term _*substrate*_ has always felt frustratingly vague. In trying to define it for myself, I took a look at Wikipedia. A clearer picture starts to emerge:

- "the natural environment in which an organism lives"
- "the surface or medium on which an organism grows"
- "the material on which a process is conducted"
- "a stratum on which another geologic stratum lies"

This tells me that if we call something a _*substrate*_, a reasonable follow-up question is: _*for what*_? In these definitions at least, the meaning of substrate is closely tied to a specific purpose or process.

I find this notion clarifying. Maybe it's not that interesting to ask, in isolation, whether something is or isn't a (computational) substrate; instead, when we say something is a substrate, we should also discuss the purpose or process.

What values guide our research?

If there is going to be a field of "substrate studies" -- as Jonathan has suggested -- I'd propose that we keep the _*specific purpose or process*_ at the forefront. As opposed to many PL venues where the context of use is either ignored, or is of secondary importance, I'd suggest that the context and purpose be critical to the discussion of substrates.

If a substrate is a particular set of primitives, chosen for a particular purpose, and offering a certain set of affordances, then the research should answer questions like:

- What is the purpose or context?
- Why was that particular set of primitives chosen?
- What are the tradeoffs?
- What kinds of use cases are well-served, and which ones are not?

In short, I'm arguing for a certain type of research contribution, which in many fields is called a "systems paper". The communities that I'm familiar with (HCI, PL, Graphics) have slightly different ideas of what good systems research looks like, but I'm fond of Kayvon Fatahalian's perspective in What Makes a (Graphics) Systems Paper Beautiful -- which I think applies just as well to other areas:

> *Good systems thinking involves establishing defensible answers to questions such as:*
>
> ** Am I convinced the work is based on a compelling set of goals and constraints?*
> ** What is the central insight or organizing principle being proposed?*
> ** What are the benefits the system provides its users? (Do I agree they are valuable?)*
> ** Can I think of alternative (e.g., simpler) solutions that might be a preferred way to meet the stated goals and constraints?*
> ** Am I convinced the design decisions made are responsible for successfully achieving the stated goals?*
> ** Does the system provide the community with a new capability that was not possible (or too difficult) to do before? What are the implications of that capability?*

Fatahalian also observes that "many good systems papers put forth a novel organizing observation about the structure of a problem."

How do we relate to other areas of CS, and other venues?

Another question worth discussing is how this workshop relates to other venues -- and other subfields of CS. At the outset, I couldn't have easily answered this question, but the process of writing this vision statement has helped me form a clearer picture. (Of course, this is just one interpretation, but hopefully it's a useful contribution to the discussion.)

- We employ ideas, approaches, and methods from diverse areas of CS, including programming languages, databases, operating systems, and HCI.
- We emphasize exploration through the construction of -- or experimentation *_in_* -- working systems. We are interested the holistic characteristics of the whole system more than in individual, isolated concepts and methods. (?)
- Our work tends to address personal- and workgroup-scale concerns, rather than "web-scale".

What else?

I suspect that Kay and Goldberg's Personal Dynamic Media is probably also a touchstone for many in the group. That paper set out an inspiring vision:

* "A programming and problem-solving tool…an interactive memory for the storage and manipulation of data…a medium for expression"
* "a dynamic medium for creative thought…allowing an owner to choose his own ways to view information. […] A machine…designed in a way that *_any_* owner could mold and channel its power to his own needs"
* "Simulation is the central notion"
* "For educators…a new world limited only by their imagination and ingenuity. They could use it to show complex historical inter-relationships in ways not possible with static linear books. Mathematics could become a living language in which children could cause exciting things to happen. Laboratory experiments and simulations too expensive or difficult to prepare could easily be demonstrated."
* "If the projected audience is to be 'everyone', is it possible to make the Dynabook generally useful, or will it collapse under the weight of trying to be too many different tools for too many people?"
* "We would like the Dynabook to have the flexibility and generality of [paper and clay], combined with the power of [cars and television sets]."
* "exceptional freedom of access so kids can spend a lot of time probing for details, searching for a personal key to understanding processes they use daily"

Perhaps it would be interesting to discuss this paper (or others) in the group -- in what ways has this vision has already been achieved, and how existing tools fall short. Also, what did the authors miss? Are there aspects of the vision that we disagree with?

It might also be illuminating to discuss the ways in which our research program(s) -- either individually or as a group -- align with the vision described in that paper (or others).

My own research

My own recent research aligns with many of the educational aims described in *_Personal Dynamic Media_*. More directly, it's inspired by the original vision of Etoys: "a media-rich authoring environment with a simple powerful scripted object model", and informed by my experiences teaching Scratch to 7- to 10-year-olds at my kids' school.

Compared to Etoys, I'm aiming for something that is more like a *_tool_* than a *_toy_* -- providing a smooth glide path to "real programming", not something that you outgrow.

Compared to Scratch:
* It shouldn't feel foreign to serious programmers. It should align with the best parts of existing computing culture, not be its own counterculture.
* You should be able to "open the hood" and understand how the building blocks are made.
* It should have a high ceiling: think Excel or Matlab. Maybe not a general-purpose programming environment, but a great tool for many serious tasks.

********** PAPER 14 **********
AUTHORS: Patrick Dubroy
TITLE: Substrates '25 Vision Statement

++++++++++ REVIEW 1 (Gilad Bracha) +++++++++
This paper is rather meta/high-level and so I'll use this review as an excuse to throw in observations on that level, based on reading all the submissions.

This field, by its nature, is feline: it attracts unherdables (aka cats). So what I expect to come out of the workshop is not some agreed/prescriptive outline of the field, nut rather a very broad delineation of approaches:

* Pragmatists who want to use, say, JavasScript, JSON, Markdown or HTML, or maybe something polyglot vs Idealists, who may not want any conventional language, or just the beautiful one(s) of their choice.

* Documents (with spreadsheets and/or computational notebooks as a special case) vs. lower-level stuff

* Servers vs. local-first

* The role of AI

++++++++++ REVIEW 3 (Tomas Petricek) +++++++++
The vision statement opens with reflections on "what is a substrate" - this takes a broader perspective than many of the other statements and raises a good question of "for what" are the substrates created (some of the other vision statements may suggest possible answers including naïve users, personal tools, collaborative work, etc.).

I'm very interested in the methodological questions raised by the paper - and this is something that I also wrote about in a short joint paper [1]. The reference to "What Makes a (Graphical) Systems Paper Beautiful" is a great pointer (alongside with other references mentioned in the UIST Author Guide [2]). If we want to turn research on programming substrates into a "normal science", we need to establish shared research methodology - and I think finding this should be one of the objectives of the meeting.

I also wholeheartedly agree with the idea that we need papers that "put forth a novel organizing observation about the structure of a problem" - alas, those are often perhaps the hardest one to write and get published because of the difficult evaluation - something our community should be aware of (and be ready for?)

The "Personal Dynamic Media" is certainly a central paper referenced in other vision statements too. However, I would be curious in close examination of the paper - why has the vision outlined in the paper largely failed to materialize? (And what do we do to avoid the same fate?)

The research project sketched towards the end of the statement sounds fascinating - in many ways, it seems that it shares some of the ideas and motivations with LOGO [3] (and since Scratch is successor to LOGO in some way, it is curious that, as happens frequently, the successful descendant loses many of the great characteristics of the original system - I would love to know how and why…).

[1] Edwards, J., Kell, S., Petricek, T., & Church, L. Evaluating programming systems design. PPIG 2019: https://tomasp.net/academic/papers/evaluating-systems/ppig19.pdf
[2] https://uist.acm.org/2025/author-guide/

[3] Solomon, Cynthia, Brian Harvey, Ken Kahn, Henry Lieberman, Mark L. Miller, Margaret Minsky, Artemis Papert, and Brian Silverman. "History of logo." Proceedings of the ACM on Programming Languages 4, no. HOPL (2020): 1-66.

++++++++++ REVIEW 4 (Clemens Nylandsted Klokmose) +++++++++
I think these are exactly the right questions to be asking. Not what a substrate is, but for what a substrate is.
I really liked the discussion about how we relate to other areas of computer science. We need to talk about what kind of community we are, and how we can avoid unintentionally alienating people from it. I couldn't help but notice that all the authors of the vision statements are male computer scientists.
I've long been inspired by Personal Dynamic Media, but just from the quotes you included, I can already tell I need to read it again.
Every time Etoys is mentioned, I feel a pang of guilt. When I first got into this kind of work, Etoys was already 10–15 years old. I struggled to see what it was about and never really looked into it properly. Maybe it's time I gave it another chance.

++++++++++ REVIEW 5 (Steven Githens) +++++++++
I really like this vision statement, for two major reasons. The emphasis on searching for the actual purpose of what we're building, especially for use cases outside the PL community, and then also for the comparisons to natural substrates and that they are meant to be places for organisms to take root and live our their lives.

We should continue to think carefully, and look outside our community for how computational substrates can serve as a medium on which people are able to grow their lives and communities. Our substrate should serve to enrich their lives, rather than exist for it's own benefit.

++++++++++ REVIEW 6 (Antranig Basman) +++++++++
Thanks Patrick, this statement sounds an important note which hasn't received enough attention in our community yet. The previous version of my vision slide https://docs.google.com/presentation/d/1Qn42eK9vkpFikDM-yU7ytMVeZTrCaXPz0clQOdE0NUY/edit asked - who might want substrates? It's very hard for our kind, I think, to avoid pursuing a categorical imperative. Having found an abstraction which could be polished or a value which can be sharpened, we can no more resist it than a drunkard can keep out of the pub.

I really appreciated the Fatahalian paper and need to make sure when I write up my substrate I organise it around his 6 questions - if I can. The systems/language split you mention is a crucial one and reminds me of Richard Gabriel's "Structure of a Programming Language Revolution" which you doubtless know well, describing how this split happened. The inability of modern communities with divergent values (e.g. those pursuing abstractions, and those serving particular communities outside themselves) to stay in contact is perhaps the biggest problem we face, which hopefully this workshop can do a little about.

In terms of this split I am sorry to report on some things from my own experience. Years ago I had a marvellous colleague (long since left the field) who remarked to me, "shit code makes for a great community". This dismayed me at the time, and still does, but he seems to have some facts on his side. The majority of my professional work these days is assisting naturalists working with R which has to be one of the most disorderly and disagreeable programming environments ever made. However it is hard to ignore the conclusion that its extreme vigour seems actually positively correlated with its awfulness - that having the PL folks out of the room is a correlate of getting stuff done that people want (a correlation I say, not quite a direct implication).

Fatahalian sets an extremely high bar, which I wonder whether many in the programming systems community can meet or have met. Am I imagining that there is something particularly clear cut about the connection of graphics programming techniques with the goals of animators etc. or are we just particularly bad at evaluating and justifying our systems? Perhaps parts of the problem are the extreme inertia of programming systems, the extreme attachment which many acquire for their favorites, and the large number of design variables which necessarily have to be tweaked at once. You have probably run into this systems survey paper by several of the workshop members “Technical Dimensions of Programming Systems (2023, Jakubovic et al) which calls out to an earlier “Programming Systems Deserve a Theory Too” which does a great job of sifting out how systems can differ from one another but says little about how they become justified and linked to the aims of a community - however it does feature a dimension of “sociability”.

Reading Fatahalian in more detail runs into this interesting point that reviewers have to be positively badgered into letting authors retain details of what communities they serve and in what context. This is an important deficiency in the “scientific paradigm” that still continues to insist that knowledge is an unsituated quantity.

A really interesting point of comparison is Boxer which in contrast to many programming systems was extremely explicit about its community and aims - its purpose was to teach elementary physics and mathematical concepts to children of about the 6th grade level. Of systems I can think of it comes closest to meeting Fatahalian’s bar, although this was done at the cost of Andy diSessa essentially divorcing himself from both the PL and programming systems communities. Andy’s answers to Fatahalian’s first three questions are extremely strong, although it was often questioned relative to the 4th and other points whether a simpler or different system could have met its goals well. Proving a negative is extremely difficult in any case.

I’m excited to hear about your own work building an environment going beyond EToys/Scratch, and I can wholeheartedly agree with the 3 points at the end of your vision which are qualities I am trying to bring to my own substrate. I’d also love to talk over how Kay and Goldberg failed to achieve their vision for Personal Dynamic Media, behind which I feel there are a number of pretty clear reasons. Members of this workshop for a few years enjoyed a reading group talking over similar papers which has now sadly lapsed.