Substrates for Naive Users

Steven Githens
diSessa Family Foundation

(2025-04-21 I apologize for the lack of references/links, as I'm already submitting this a day late. Planning on adding them soon)

My software engineering career has largely been involved in working with multidisciplinary teams in areas such as online learning platforms, medical research teams and homegrown EMR's in East Africa HIV treatment, breast cancer biobanks, and accessible computing systems. Most recently I've been working on modernizing the historical Boxer codebase and preparing it for the next generation. As a result of this I'm primarily interested in helping naive users accomplish daily tasks for life and learning, as well as build interesting new customizations on top of them. My general litmus test is, "Could someone who doesn't know what Markdown is use this?"

In general, I largely agree with Jonathon Edwards six bullet points for what is a substrate. And other papers in this track have done a great job of summarizing various implementation details about what is necessary to build the infrastructure for a substrate to achieve these aims. So I'll just list out some general ideas I have for Substrates (at the moment heavily influenced by my work on Boxer), and general quandaries I'd love to discuss and co-operate with folks on.

The Search for the Ultimate Creative Note Taking Tool

Like many computer scientists and software engineers, I've spent countless hours of my life (which I'll never get back!) to find the ultimate note taking and organization tool… my first Palm Pilot, Emacs/Org Mode, Evernote, Chandler, various configurations of MS Office with insane COM embedding/linking setups, DB toolings, fancy Django Admin consoles, ipython/jupyter notebooks, Ink&Switch prototypes, whatever hot new Markdown based tool was posted on Hacker news this week, ad infinitum.

I think this exotic search for a paradise of note taking still influences many of us and the needs for a Substrate. It should be possible to build these types of applications on the Substrate. Note taking, journaling, and planning are pretty common tasks for the people of this Earth.

Disclosable Computing and gradients between editing and viewing

Somewhat related to Antranig Basmans discussion of Disclosable Computing, I'm very interested the various levels of editing/building a document, vs viewing/playing a document. For instance, if I'm building a physics lab as a Boxer Microworld, I usually want users to interact with it through a specified set of controls rather than being able to muck around with the implementation willy nilly. Similarly, if I want to view a document built in a Substrate as a webpage or readable document (similar to a PDF) there needs to be a clear way to switch between these.

I believe there is a lot of useful work we can do on this as a Substrates group. Having worked

with friends trying to build their websites with WiX, Square space, or similarly soul crushing user interfaces, the bar can only go up.

Natural Language as a Gateway Drug

One of my favorite things about Boxer is that it starts out as a Word processor, and then you can discover computational aspects of it later on. Perhaps just by doing some inline math (rather than going to the calculator app), or being able to use new primitives to accomplish a task. It turns out Humans are usually fairly comfortable expressing things in their native natural language. If they can start there and add in computational expressions later, I think it could be quite an empowering transition.

Looking to Simplicity from Open World Substrates in the video game sector

I think it would be very useful to spend more time looking to non-textual segments of the creative industry to gain some useful feedback. Applications such as AutoCAD and Blender, and video "games" such as Minecraft or Nintendo's Switch era entries in the Legend of Zelda Franchise, present compelling open world experiences for UI such as navigation, discoverability, and building. When I see kids building chip fabrication plants in Minecraft, I realize that it's probably one of the more successful Substrates I've seen.

Based on my work in Boxer, I think it's important to have an open world experience. Users should be able to build, nest, or insert new creative content wherever they want. But often whats lacking is the UI that makes this manageable and fun.

Furthermore, I look to the games industry as an example of how packaging should work and how deployment should be streamlined. Some of my worst experiences in software development have been package managers (such as Maven, NPM, and Quicklisp), or even packaging environments such as Docker and Kubernetes. These all promise a plug and play development environment, until you barely step out of the confines of normal usage. As builders of substrates, we should be sure not to pass these errors on, and consider it a huge failure on our part if our users ever see some asinine error about "Missing dependency".

Accomplishing basic daily tasks

Journals, lab notebooks, documents, publishing. I want complex computational ideas and development to be possible in our substrates. But first and foremost, I want basic documents, the kinds people use to capture inspiration and creativity to be easy to create in the Substrate and easy to share via whatever routes. I want it to be fun and reasonable to create large scale open world simulations and environments.

I would be remiss in the year 2025 not to insert something about AI here. I'm fairly ambivalent on the topic to be honest. I think real humans and artificial ones should be able to author documents and worlds in the Substrate. I would love to have an AI in the vein of Marvin the Paranoid Android say things like "Why stop creating templates for your blog posts, just when I'm hating it." But what I really hope, is that a compelling Substrate is something that people will use to investigate a new physics principle, or journal about how their beach walks change depending on the tide and phase of the moon. That the journey of their usage is more

important than the finished product itself, and they'll relish in the creative spirit of humanity while using the Substrate to flesh out that vision. Our mission couldn't be more important.

********** PAPER 12 **********
AUTHORS: Steven Githens
TITLE: Substrates for Naive Users

++++++++++ REVIEW 1 (Gilad Bracha) +++++++++
A few comments (I'm in general agreement).

On "Disclosable Computing". This a very real issue. One option is to make (sub)documents read-only. In particular, UI elements, themselves recursively composed of smaller ones, can default to read only. Another aspect is how to manage the discoverability of meta-controls. I often have an "inspect presenter" option in menus that allows you to manipulate a UI component as an editable object. Collapsible elements are useful as well - but they all take some real estate. I would like to experiment with other approaches, like hiding meta-functionality inside help text.

Re: "Natural Language as a Gateway Drug". This is where AI comes in. Ways of giving instructions in the editor targeted at the AI, and then opening up richer prompting tools (e.g., Erik Meijer's work on Universalis) either explicitly, or by addressing the AI in the text (a textual version of "Hey Siri").

Packaging and deployment technologies must be zero-install (I use web pages) or at least plug-and-play. Nothing should ever need to be "built". Documents should be self contained (I use zip files (ugh) because in that respect, the web is painful).

Games are probably a great inspiration. I think we can expect a move from 2D documents to 3D, VR & AR.

++++++++++ REVIEW 3 (Tomas Petricek) +++++++++
The vision statement is perhaps unique in that it focuses on "naïve users" - the question whether substrate can be used by someone "who doesn't know what Markdown is" is a very good litmus test for what a substrate enables! The statement emphasizes what I would call humanistic values - including creativity, inspiration and sharing. I also think that personal notetaking, mentioned in the statement, is a good use case for research projects around substrates.

The statement notes that you can start using Boxer as a word processor and then gradually learn - this is an important design principle for substrates (shared by many submissions and perhaps even necessary?) - I always found interesting the discussion about this characteristic of Smalltalk in [1]; it may also be useful to relate this to HyperCard "user levels".

The paper also suggests we should see Minecraft as a substrate - this is a great point and I think potentially a very illuminating example for the work that the group will do in the future!

[1] Trygve Reenskaug. User-Oriented Descriptions of Smalltalk. BYTE, 6 (8). 1981.

++++++++++ REVIEW 4 (Clemens Nylandsted Klokmose) +++++++++
I like how Steven sees this search for the ultimate creative note taking tool as a symptom of a lack of a proper substrate.
“One of my favorite things about Boxer is that it starts out as a Word processor” I think this is really important. We need to start with stuff, then we can add interaction and dynamism later.
Also, very good point that we should remember to consider something like Minecraft as a pretty successful substrate!

Like Steven, I am also ambivalent about AI. I have come to see that it won’t go away, so we should probably figure out how to embrace it somehow.