UnitedLex “Idea Factory” - Computing Lab

Mandate

At UnitedLex we are building a lab in the spirit of Xerox PARC with the broad mandate of creating "the information office and computing systems of the future." PARC was instrumental not only in creating foundational technologies that revolutionized computing, but also in shaping modes of thinking about these technologies. Our intention is to foster a similar environment of innovation and creativity. The outlook is for the long-term. As opposed to the quarterly profits or incremental feature wrangling, the thinking will encompass the way in which human interaction with technology and information can be shaped in the next 5-10+ years. Our goal is to shape the future, to invent it, or at the least to resurrect some of the great ideas of computing lost in the antic hay of commercialization over the last four decades.

We are not married to the largely ill-defined boundaries of software or hardware, front-end or back-end, low level or high. These distinctions have no real place in our thinking, and perhaps some of these have no place in the technology world at large.

One such distinction will be an explicit target of our research: the strict divide between computer “users” and “programmers.” We take as a basic tenet that computing is not a tool, but a medium for literacy-like interaction. Our goal is to explore system-design in these terms, with the desire to allow literate expression between users and computers. In continuation of the literacy metaphor, we won’t expect everyone to write like Joyce, but we do expect anyone who is interested enough to be able to author in the medium of computing. What “authoring” means here will, in part, bear out in the course of our investigations.

We will stay conscious and aware of the business and commercial immediacies of UnitedLex and its partners, but we will not limit ourselves to a thinking circumscribed by these realities. Under the proper guidance everything we create will have large-scale market potential, but our metric for project worthiness and broader mandate will be manifold.

Problem Landscape

Our problem landscape will primarily focus on the following:

- Information: contextual, actionable, describable, and naturally recursive
- Entities that derive new information (knowledge) from current information: human, machine, networks, or otherwise
- Framework(s) that support this dynamic: hardware, software, broader infrastructure, and cost

We see the following global problems with current approaches to personal computer technologies:

- Systems are not expressive, i.e. environments do not correspond to the flow of expression that is natural to the medium of work
- Systems are not extensible or malleable, i.e. the environments are not natively configurable to the individual flow
- Systems are not comprehensible, i.e. they are so large in code and complexity that regardless of technical ability no single individual is able to understand the systems bottom-up
- Systems are not pedagogical, i.e. they do not constitute environments in which users can actively learn how to manipulate them, and they are not "examples of themselves"

The byproduct are solutions which are

- Inherently static from the perspective of the individual
- Inherently static from the perspective of infrastructure and support
- Grow to colossal pyramids of complexity and code
- Incur colossal cost in both time and resources

Let us illustrate these issues with a few simple examples.

Consider the button in a user interface. Buttons do all sorts of things on our modern systems, from starting a download to making text bold. All of us have deep experience with buttons. We are so attuned to their usefulness and value, that we almost don't even think about it consciously. But there is a problem: what if you want to "make your own button" to do something? Here you run into a short and sobering list of options. You could find or buy software that already tries to do what you want, fighting within its constraints. Or you could learn a full-fledged programming language, along with the corresponding development environment, related libraries, and intricacies of the given operating system. What should be an easy, high-level, almost natural thing to do suddenly becomes extremely difficult. This is an egregious oversight, since the whole point of computing is that "it can be anything." This oversight has wider consequences too, in that users come to expect what computers can or can't do when they are at the helm.

Let's take an example more specific to e-discovery and document processing, core in UnitedLex's business initiatves. Imagine you are ingesting information from a set of documents, and the current software does not include a key field. You are able to

indicate which field you want, and how you would like this text extracted and processed. You can even indicate all this in a few gestures using the keyboard or mouse but how do you go about doing so? How do you ask the software to update itself?

Again what should be easy and intuitive, something you can describe to anyone with a few examples becomes a deep-dive engineering and programming exercise. And if you are not the engineer aware of how to make the necessary changes you are limited to relying on new or updated software. These "solutions" are at best panacea and at worst critical failures.

And as a final example let us take a "lower-level" design decision that has had fundamental consequences in both infrastructure, interface and has shaped not only how software is created, but entire industries. Consider the SQL database. It is the most popular database type to date and is the foundational software layer of information driven sectors such as health, finance, records and compliance, libraries, and again e-discovery. The interface to the SQL database is the SQL query, a deterministic amalgam of boolean conditions which define a subset of information, i.e. either an information entity satisfies the condition or it does not. Over the last two decades much time and energy has been spent optimizing and scaling the performances of such stores and adding a continual stream of features such as basic NLP functionalities for text heavy fields.

But when is deterministic lookup an effective interface to information? Outside of inventory, when do people think and work deterministically? When has this been an effective system of exploration and knowledge in the history of human thought? Our internal processing is inherently associative, potentially modelled by Baysian inference or a statistical framework in that spirit. We don't think deterministically, we don't learn this way, we don't discover this way. But legal professionals submit lists of keywords to judges for approval, risk officers decide on what information to store and what to destroy based on boolean filters, historians are forced to discover the past one query slice at a time. The bifurcation of the natural thought process from the software flow is startling.

The pioneers of personal computing made many of these same observations in their own time. For example, Douglas Engelbart's NLS had associative indexing built-in at the OS level -- it was *not a software feature*, but a fundamental component of the system itself. Other systems like Sketchpad, Smalltalk, Hypercard, and more demonstrated holistic and effective approaches to some of these core problems, even though their lessons are not mainstream today.

We believe that these historical examples show that a different kind of computing is possible and desirable. We also believe that they relied upon two architectural components (which are largely absent from modern approaches):

- Metaphor - a unifying semantic and semiotic landscape for architectural design of the solution
- Design - the concise and elegant instance that is the implementation of the metaphor into system

The above two components are critical to ensure that what we create is as simple as possible, but not simpler: they define the boundaries of what is deemed as sufficient and necessary in any given solution.

We have three advantages over the pioneers in computing: historical precedent and examples, universal modern communication standards (HTTP, TCP-IP, etc), and the flexibility and relatively low cost of hardware. We believe these provide an unprecedented opportunity.

To use another metaphor, without the invention of the simple mechanical lever we are forever limited by raw physical power. And without this invention it is impossible to imagine an infinite level, i.e. the wheel, which not only allows us to lift colossal weight but also to transport it over distance. From there we can imagine the flywheel, the perpetual motion machine, and so on.

Our primary goal is to design solutions today which will allow us to imagine those of the future.

Structure/Framework

The organizational structure of the lab will be flat and any member will have the right to propose a new or join an existing project. Every project must have a leader. This individual will be responsible for project management and reporting. This leader will also be expected to explain, or evangelize, the core ideas to the lab and UnitedLex personnel at large as called for.

Lab members will have the option to work individually or collaboratively, with the latter being favored over the long-term. Although given the small size of the lab at the moment a single overarching project will be chosen with the entire team contributing.

The lab will meet (virtually) one time per week in an open-ended session to present ideas, gather feedback and discuss. Every such meeting will begin with a 15-30min presentation by a designated lab member who will report on progress, run a small informational session or pitch a new project.

Week-long retreats will be held every 6 months required by all lab members to attend. These will provide an opportunity to work together in the same space and compensate for some of the dynamic not available with remote collaboration. The

exact structure of these retreats is to be determined at a later time. Given world-health realities, the first retreat will be held at the Pajaro Dunes Resort center (this is where many of the PARC people gathered during the 1970s).

Reading and viewing list

The lab will have a running reading/viewing list roster. The required reading will be limited to computing, the suggested can be broader. The goal is for all lab members to have gone through the required materials within the first 6 months of joining. Below is a sample:

Required:
Man-Computer Symbiosis, JCR Licklider
Doug Engelbart's 1962 Project Summary Report
Douglas Engelbart (1968)
Personal Dynamic Media, Adele Goldberg and Alan Kay
GRAIL
    wiki and report papers there in
    demo
Sketchpad (PhD Thesis), Ivan Sutherland
Sketchpad Demo, Ivan Sutherland
The Future of Reading Depends on the Future of Learning Difficult to Learn Things, Alan Kay
Mindstorms, S. Papert
Hypercard, Bill Atkinson et al (1987)
Inventing the principle, Bret Victor (2012)
Media for Thinking the Unthinkable, Bret Victor (2013)
Yesterday's Computer of Tomorrow: The Xerox Alto, Dan Ingalls (2017)
Viewpoints Research Institute, STEPS final report (2012)

Suggested:
The Evolution of Smalltalk, Dan Ingalls
The Act of Creation, Arthur Koestler
Tacit Dimension, Michael Polanyi
Science of the Artificial, H. Simon
Understanding Media, M. McLuhan (or some other McLuhan)
Orality and Literacy, Walter Ong
The Printing Revolution in Early Modern Europe, Elizabeth Eisenstein