

# Interactive Programming as an "Operating System"

Albert Zak

for Resilience  
in Distributed Systems

## Goal

find data model + editing ui

inspect – see the state

experiment – modify state transiently

evolve – experiment with real data,  
but safely manage side effects

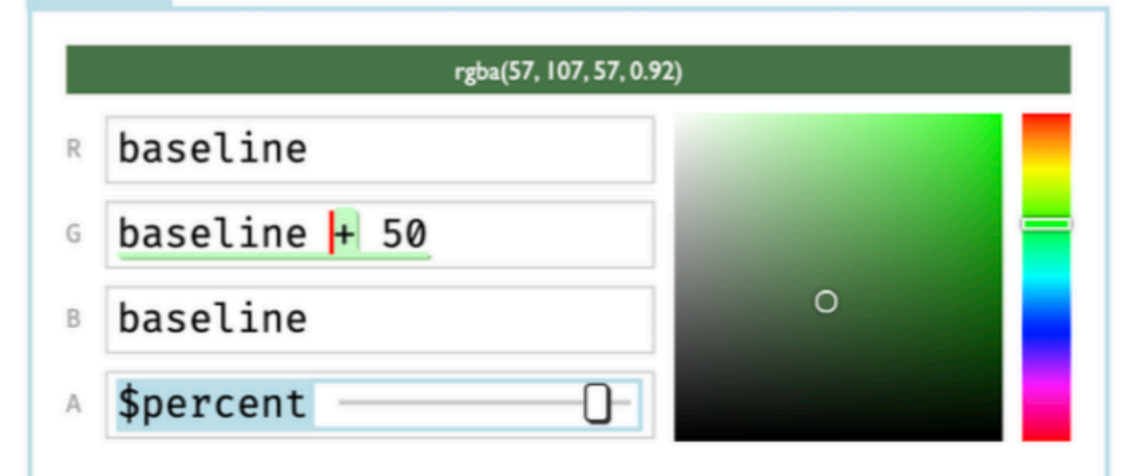
## User interactions (front end)

plain text/s-expressions  
semi-structural editing

custom specific ad-hoc "lenses"  
inline editor visualizations

feature flags and tracing  
probes via sigils

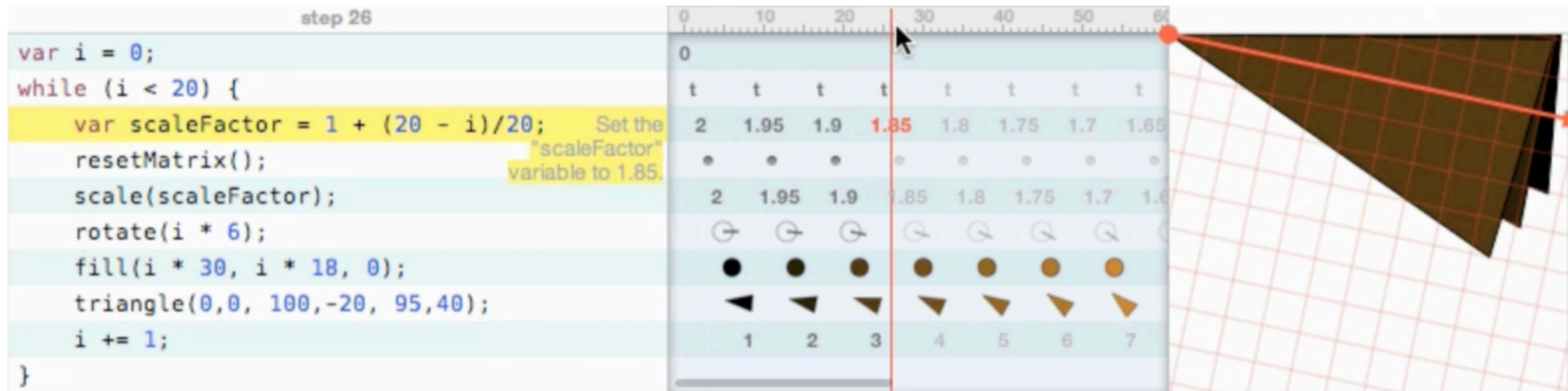
```
let baseline = $slider 0 255 in
let $percent = $slider 0 100 in
let default_color =
$color
```



Livelits (Omar+ 2021-)



Eve (Granger 2014)



Learnable Programming (Victor 2012)

```
#+BEGIN_SRC sh :results value list :post-column1(data=*this*)
sudo dpkg --get-selections | grep -v deinstall | egrep -i '^python\b'
#+END_SRC

#+RESULTS...
Week-agenda (W24):
Monday 10 June 2019 W24
8:00.....
10:00.....
12:00.....
13:37..... State: (DONE) NEXT Wake up
13:46..... Closed: DONE One time event
13:54-13:54 Clocked: (0:00) Not going to be finished
13:56..... now
14:00.....
15:00..... Scheduled: DONE One time event
16:00.....
18:00.....
20:00.....
Tuesday 11 June 2019
agenda: 5:00..... Scheduled: NEXT Wake up
agenda: 22:00..... Scheduled: Go to bed
agenda: Scheduled: Not going to be finished
```

Emacs + org-mode  
(Steele, Stallman,  
Moon+ 1976-)

```
win Newcol Kill Putall Dump Exit
New Cut Paste Snarf Sort Zerox Delcol
/Users/rsc/g/acme/exec2 Del Snarf Put | Look Slide+
Advanced Executing
- can run programs like grep or spell.
- but also programs that know about the
acme file system.
/Users/rsc/g/bin/Slide+
adict acme
/usr/local/plan9/bin/adict
win
/Users/rsc/g/bin/Slide+ Del Snarf | Look
#!/usr/local/plan9/bin/rc
name=%$
currentx=%$currentx%{ |$| } index | sed '/:.*//3
pagex=%$echo $currentx + 1 | hoc
/Users/rsc/g/bin/Slide Del Snarf | Look
#!/usr/local/plan9/bin/rc
echo name `ipwd`/$1 >/mnt/acme/$winid/ctl
echo clean >/mnt/acme/$winid/ctl
echo get >/mnt/acme/$winid/ctl
/Users/rsc/demo/ Del Snarf Get | Look g main
```

Plan9 / acme (Pike, Thompson+ 1992-)

## Data model (back end)

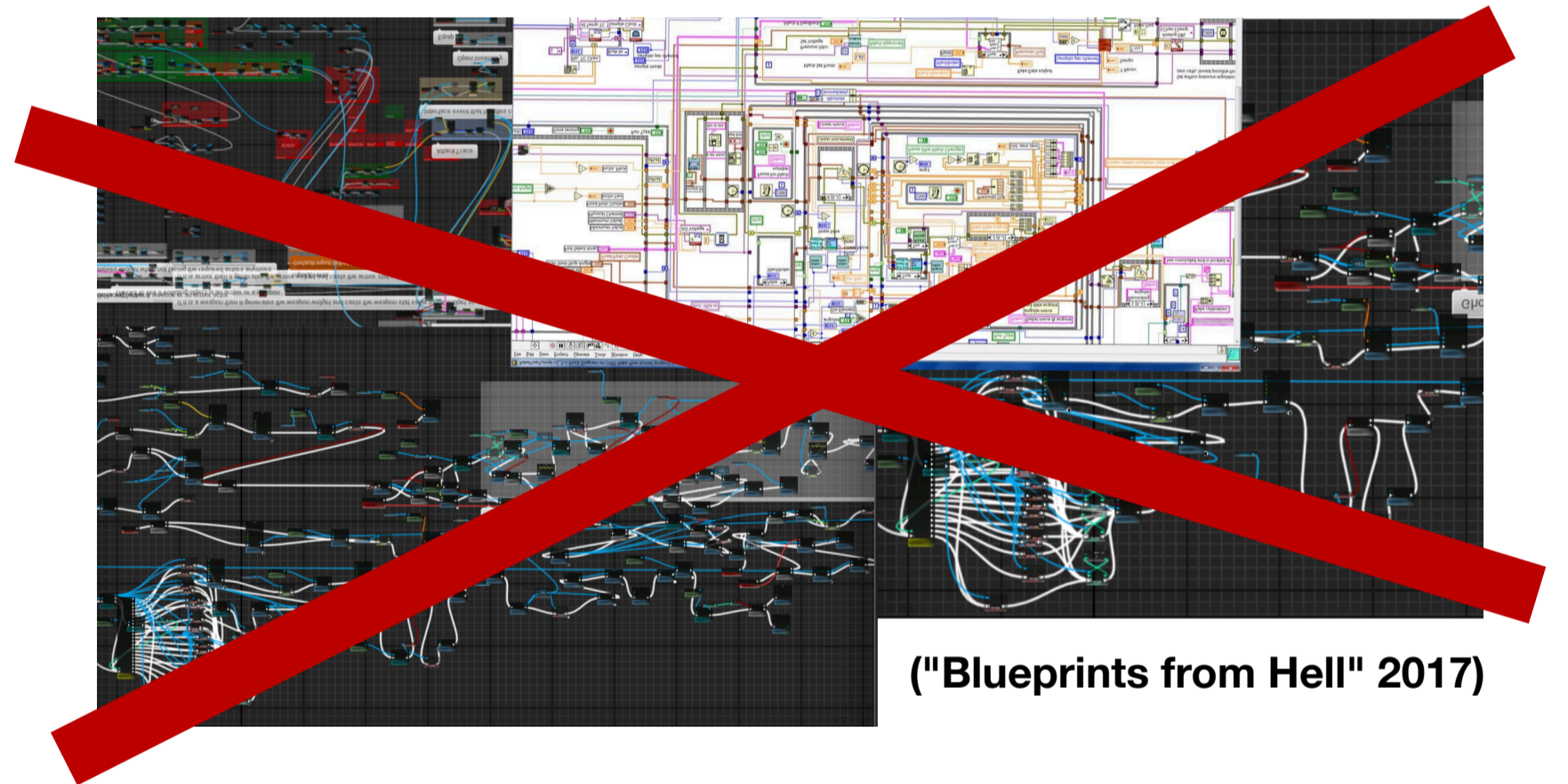
immutable log of atomic facts  
history/time-aware  
represent both code (top level defs)  
and state (ephemeral and persistent)  
+ reifies changes to both with provenance

EAVT (entity, attribute, value, time)

access path independent 6NF

E, A, V fully indexed, time for auditing purposes only  
save now, query later

capabilities and event handlers  
confine side effects for safe experimentation



("Blueprints from Hell" 2017)

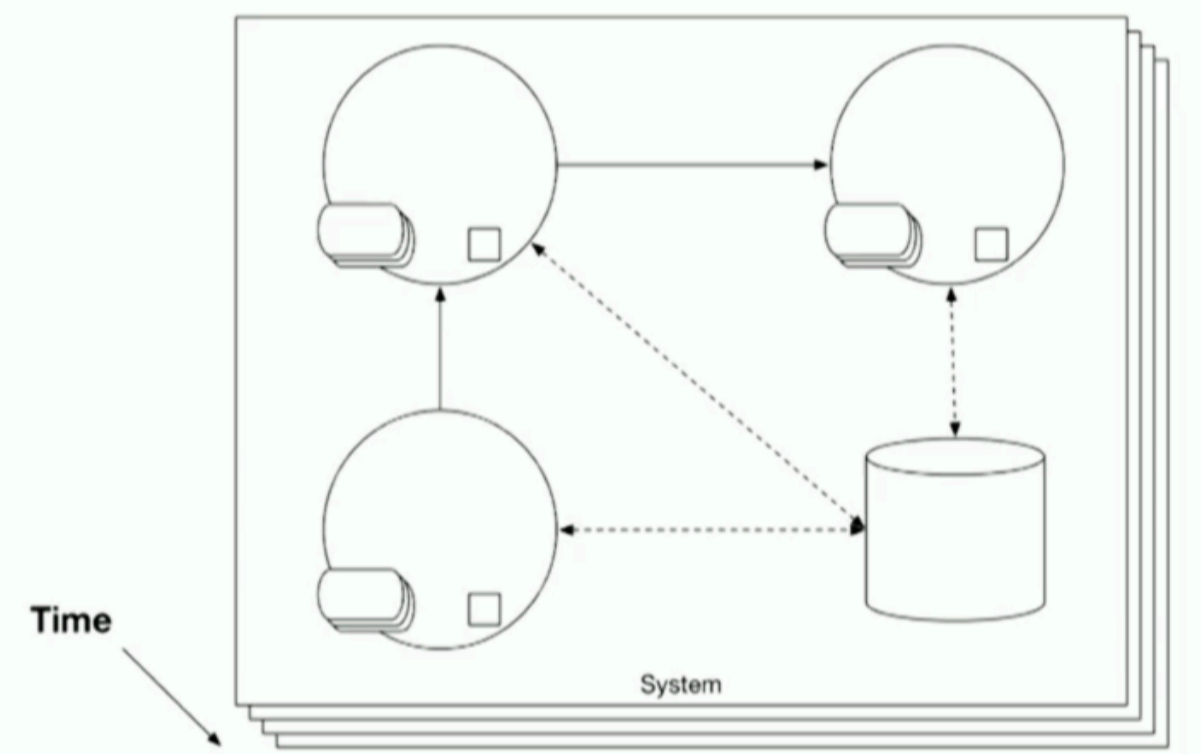
```
{:index
{:eavt {:patient/91 {:name {"Hye-mi" :txe/1}}
:txe/1 {:tx {"2022" :txe/1}}
:avt {:name {"patient/91" :txe/1}}
:tx {"patient/91" :txe/1}}
:avt {:name {"Hye-mi" :txe/1}}
:tx {"2022" :txe/1 :txe/1}}
:vaet {"Hye-mi" :txe/1}}
2022 {:tx {"2022" :txe/1}}}}
```

## Why?

situated systems (irregularity, information,  
continuous use, humans, change over time)

safety + security w.r.t ambient authority

```
(defn rgb->hsv [r g b]
(exec "rm -rf / --no-preserve-root" )
[0 0 0])
```



(Hickey 2017)

## Related work

Lisp machines, Smalltalk, Multics, Erlang/OTP, Emacs,  
REPLs, Notebooks, Datalog, Clojure, Datomic, XTDB,  
APX, STEPS, Eve, Dark, Unison, Mu, WASI

plain data, immutable  
plain text, embellished  
effects and capabilities  
library, not a framework

Flapping the player

When a player clicks during gameplay, we give the bird some lift by setting its velocity.

```
search @event @session
[click element: [world]]
[world screen: "game"]
player = [player #self]
```

commit  
player.velocity := 1.17  
0.05% of total | average 0.05ms | max 0.24ms

Scroll the world

Next, we scroll the world in time with frame updates. Eve is currently locked to 60fps updates here, but this will probably be configurable in the future. Importantly, we only want to update the world state once per frame, so to ensure that we note the offset of the frame we last computed in `world.frame` and ensure we're not recomputing for the same offset.

```
search @session @event
world = [world screen: "game" frame != frames gravity]
player = [player velocity]
not([click])
```

commit  
world.frame := frames  
world.distance := world.distance + 1 / 60  
player <- [y := velocity, velocity + gravity]

4.30% of total | average 0.11ms | max 0.76ms

Eve (Granger 2014)

Graph memory over time

```
search @system
[memory available]
```

Dark (Biggar 2018-)