"THE WAY TO EDEN"

Steve Miner
@miner

edn-format.org

edn is a system for the conveyance of *values*.
It is not a type system, and has no schemas.

Schema

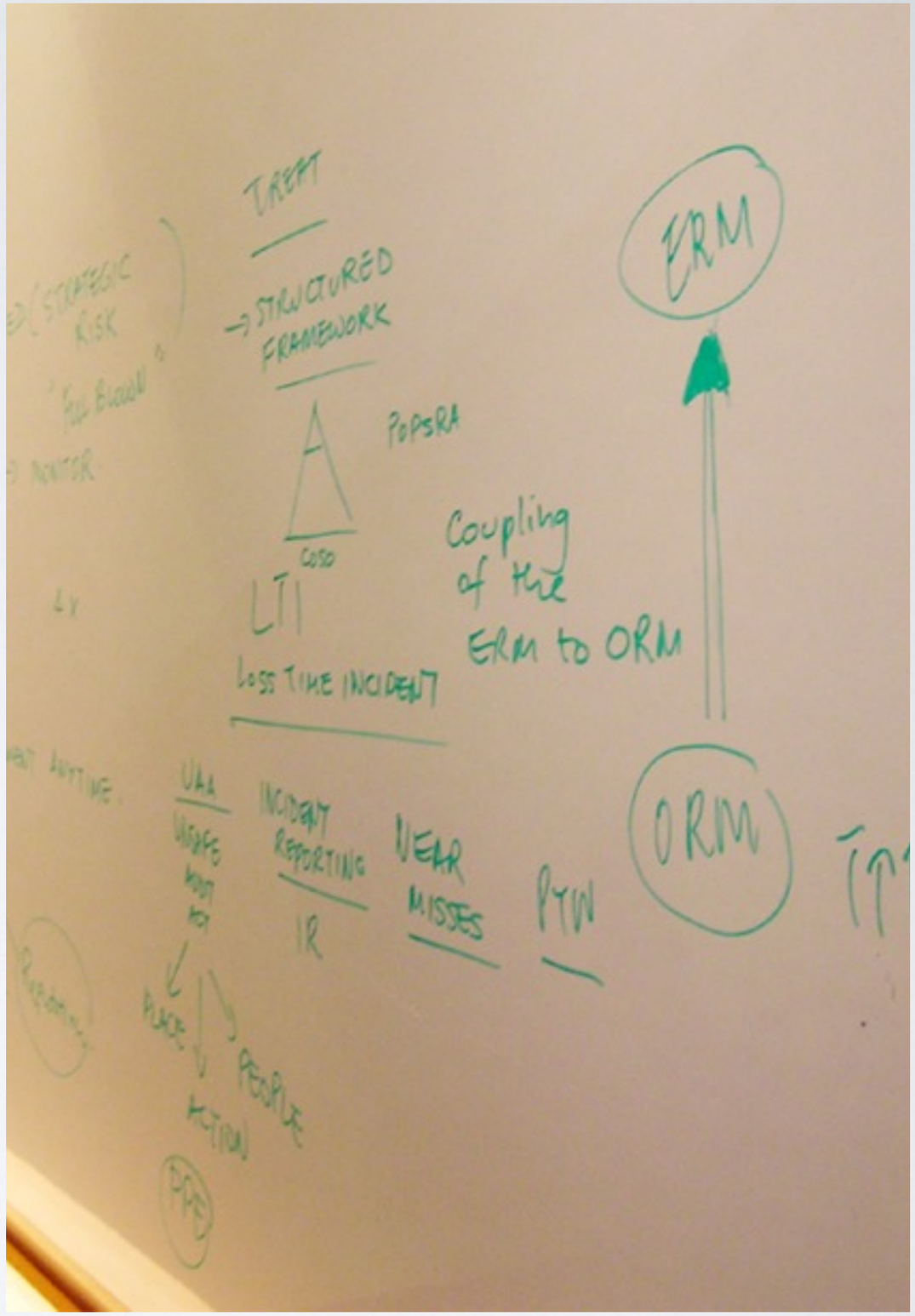
the shape of data

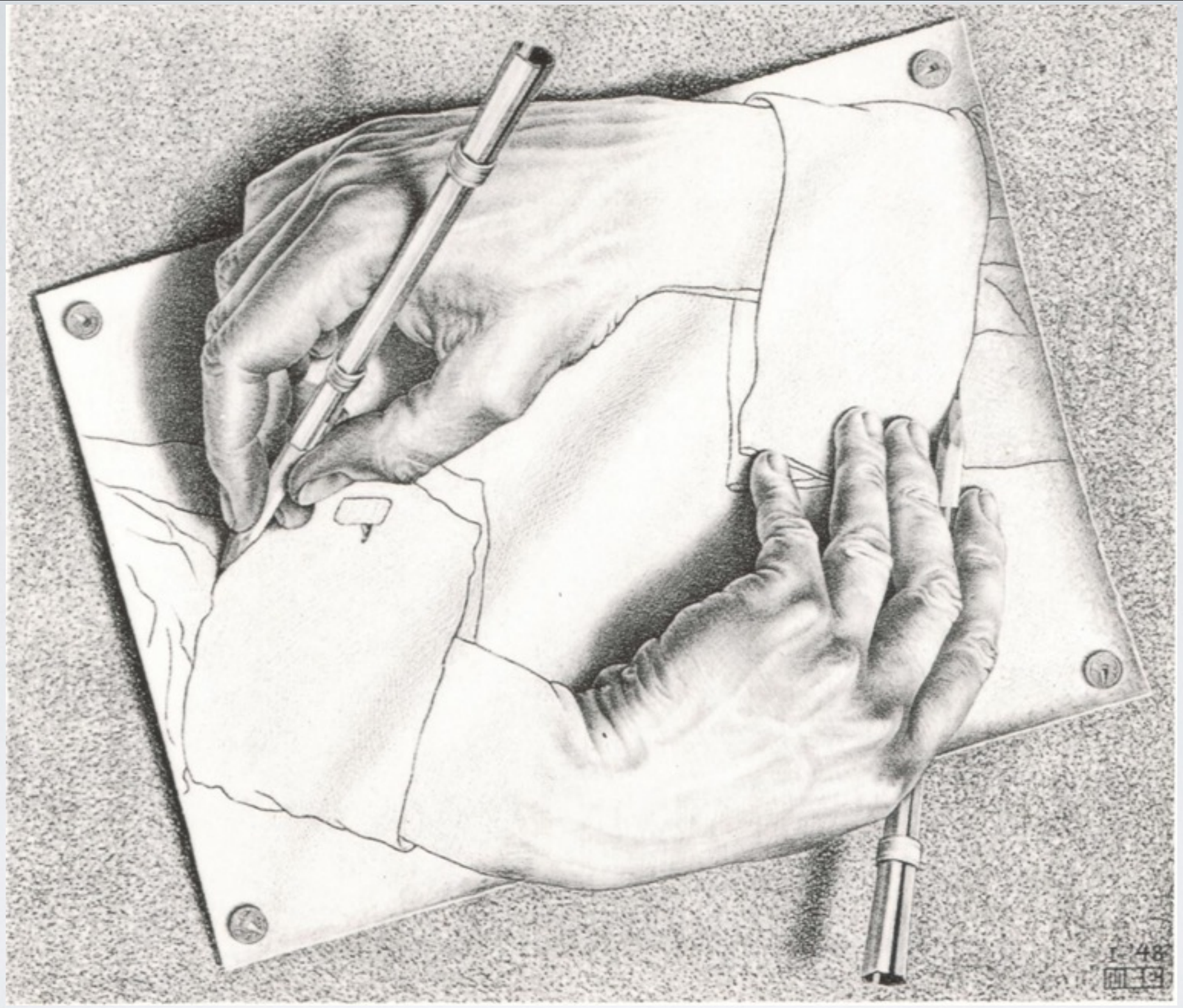
Herbert

a schema language for EDN

Whiteboard Compatible

simple
terse
readable





EDN in EDN

Patterns

- Literals - :kw, “foo”, 42, nil, true, false
- Symbols - int, kw, str, sym, list, vec
- Maps - {:a int}
- Vectors - [int kw]
- Operators - (or nil (and int (not 42))

Quantifiers

- (? kw)
- (* int)
- (+ sym)
- kw? int* sym+

{:a int :b? sym :c [str*]}

matches:

{:a 42 :b foo :c [“abc” “def”]}

{:a 42 :c [“x” “y”]}

{:a 42 :b foo :c []}

Grammar

```
(grammar [person+]
    name (and str (not (pred clojure.string/blank?)))
    handle (str "@.+")
    person {:first name :last name :twitter? handle})

[{:first "Steve" :last "Miner" :twitter "@miner"}
 {:first "James" :last "Kirk"}]

(grammar (list 'grammar pattern (* term pattern))
    term sym
    pattern any)
```

Bindings

[(:= n int) n n] matches [3 3 3]

{:a (:= s sym) :b (not s)} matches {:a foo :b bar}

[(:= low int) (:= hi int) (int* low hi)]
matches [3 7 4 6 5]

{:len (:= n int) :vals (and (cnt n) [sym*])}
matches {:len 5 :vals [a b c d e]}

Tags

- (tag inst) matches any java.util.Date, java.util.Calendar and java.sql.Timestamp

- (tag my.ns/Rec) matches any record my.ns.Rec

- (tag my.ns/Rec {:a int}) matches #my.ns.Rec{:a 42}

- protocol miner.tagged.EdnTag in **tagged**

Herbert API

(conforms? *pattern value*)

(conform *pattern*)

(def my-test? (conform '[(:= k kw) sym+]))

(my-test? '[:a foo bar baz])

;=> {k :a}

Herbert

- Open source on github and clojars
- https://github.com/miner/herbert
- https://clojars.org/com.velisco/herbert

Thanks

- Eric Normand - **squarepeg** parser
- **tagged** library - tagged records
- edn-format.org
- Prismatic **schema**
- runa-dev **clj-schema**
- http://tos.trekcore.com for Star Trek photos

Star Trek

{:title
"The Way to Eden"
:stardate
#tos/stardate 5832.3
:airdate
#inst "1969-02-21"
:season 3
:episode 20}

"THE WAY TO EDEN"

Steve Miner
@miner

Examples

- [kw str int] - [:a “foo” 4]
- {:a int :b sym :c kw} - {:a 42 :b bar :c :baz}
- (keys kw sym) - {:a foo :b bar :c baz}
- [int+ kw?] - [1 2 3 :end]

Martin Fowler

- "schemaless" means *implicit* schema
- prefer *explicit* schemas
- http://martinfowler.com/articles/schemaless/
- https://www.youtube.com/watch?v=8kotnF6hfd8

Uses for Schemata

- Documentation
- Validation
- Pattern matching
- Data transformation