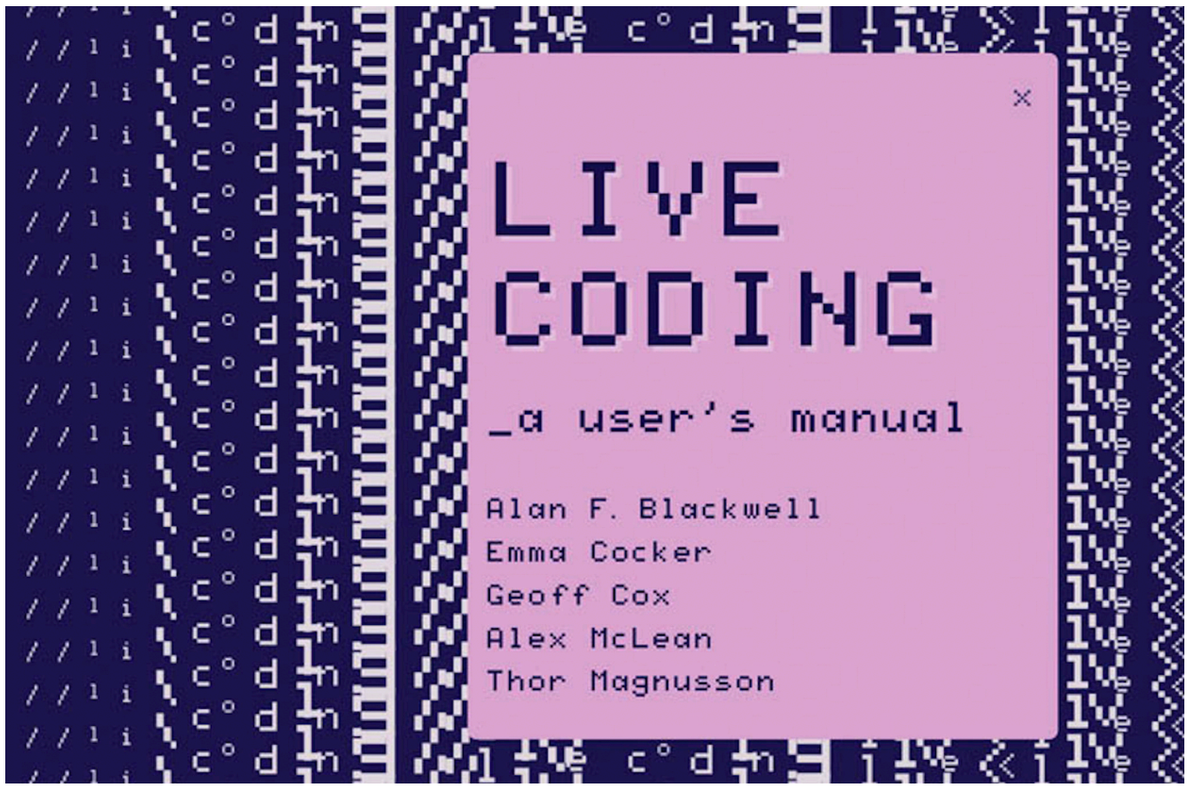
Composing and Performing Electronic Music on-the-Fly with Pharo and Coypu

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo ASSOCIATION

Live coding

In **Live coding**, sometimes referred to as *on-the-fly programming*, *just-in-time programming*, or *conversational programming*, music and video artists expose and rewire the innards of software while it generates improvised music and/or visuals.



Live coding has become increasingly popular in computer music, often as improvisation or combined with algorithmic composition.

All code manipulation is projected for the audience's pleasure

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo ASSOCIATION

TOPLAP MANIFESTO

__T__ransnational __O__rganisation for the __P__roliferation of __L__ive __A__rtistic __P__rogramming

- Give us access to the performer's mind, to the whole human instrument.
- Obscurantism is dangerous. Show us your screens.
- Programs are instruments that can change themselves
- The program is to be transcended - Artificial language is the way.
- Code should be seen as well as heard, underlying algorithms viewed as well as their visual outcome.
- Live coding is not about tools. Algorithms are thoughts. Chainsaws are tools. That's why algorithms are sometimes harder to notice than chainsaws

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo ASSOCIATION

ICLC

ICLC25, Barcelona ,Catalunya

ICLC24 - Shanghai, China - Algorave @ System

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo ASSOCIATION

Why Coypu?

Smalltalk syntax is great for people with little computer literacy.

Pharo is easy to learn → Many resources (2 MOOCs)!

Pharo is easy to learn → It''s reflective!

Pharo is easy to learn → Lively (relatively :)) young community.

Musicians and sound artists might be intimidated by functional or opaque languages

Pharo comes with its own IDE = No extensions to install

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo Association

Habitat

Coypu has been developed to program music ***on-the-fly*** with Pharo.

Coypu acts like a *score** for an *orchestra**.

* in traditional *Csound* terms

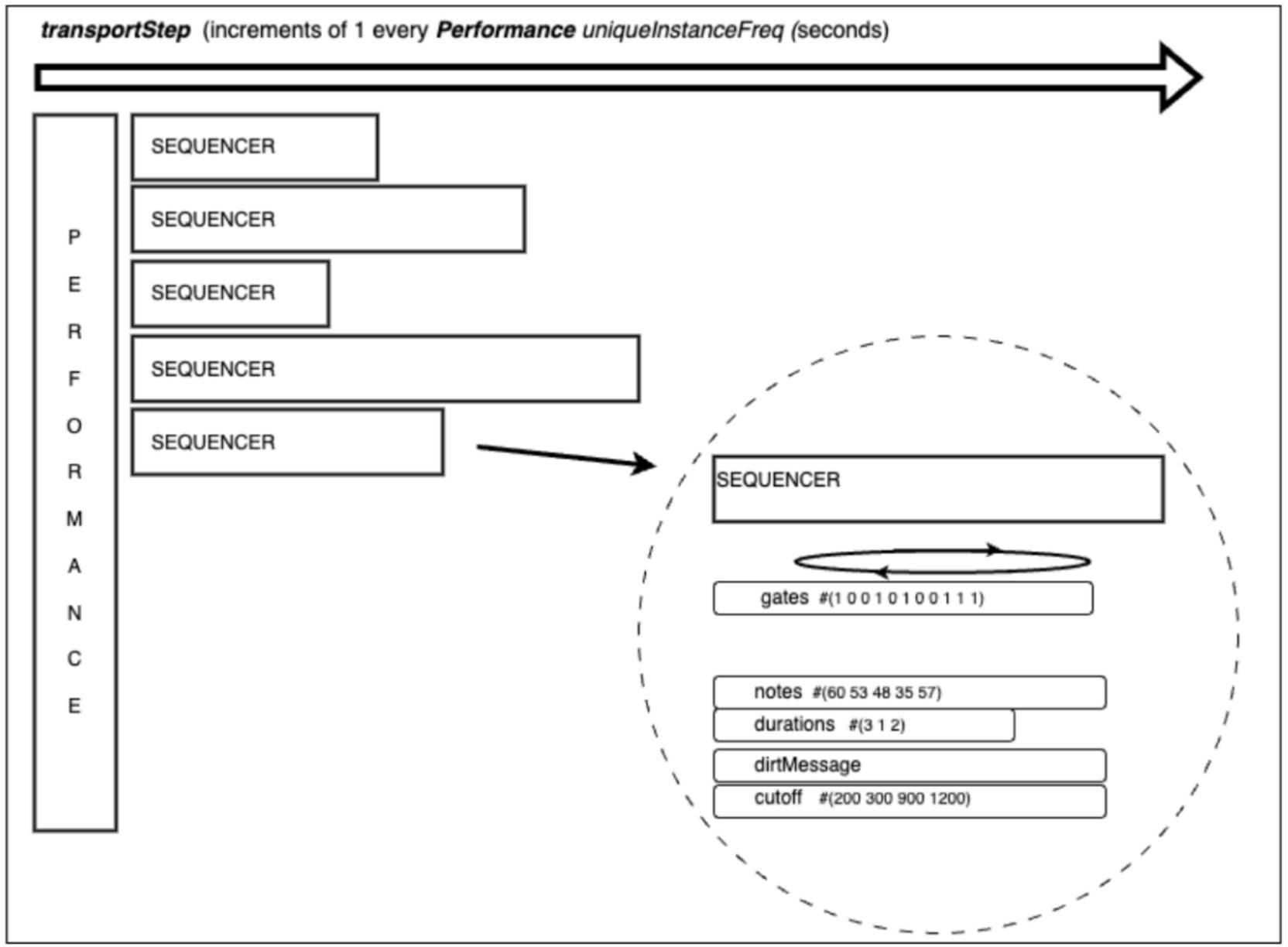
The *orchestra* is the audio server used to render the sounds

Coypu has been developed to program music ***on-the-fly*** with Pharo.

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo ASSOCIATION

Anatomy



Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo Association

Hello Coypu

```
p := Performance uniqueInstance .
p performer: PerformerPhausto new.
p freq: 98 bpm.

#rumba to: #conga.

p playFor: 256 bars.
```

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

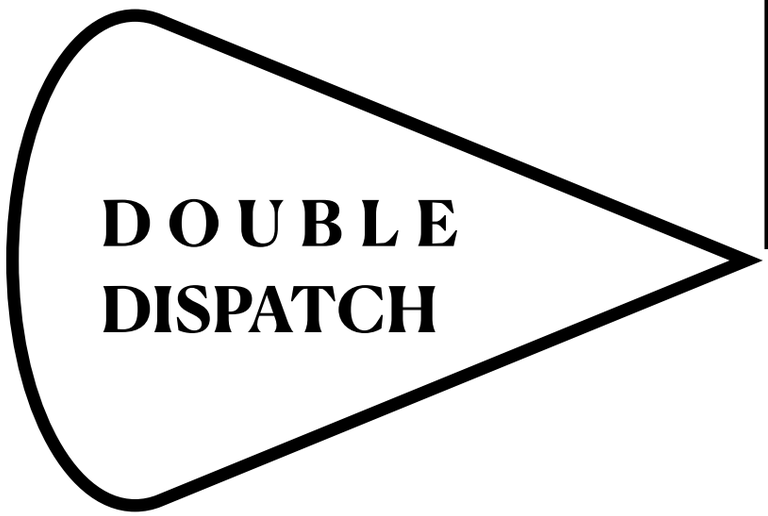
Pharo Association

One performance

The Performance is a subclass of Dictionary

Each key of the Performance is assigned to a Sequencer.

The Performance is a Singleton.




```
Performance >> playFor: aNumberOfSteps

self class performer playFor: aNumberOfSteps.
```

The Performer is responsible of
sending events to the audio client!

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

5 Performers

`PerformerLocal` → Local OSC audio server (ChucK PureData, SuperCollider, MaxMSP)

`PerformerKyma` → Symbolic Sound Kyma (external OSC connection)

`PerformerSuperDirt` → SuperDirt (local OSC connection)

`PerformerMIDI` → External MIDI hardware or local application

`PerformerPhausto` → Phausto (embedded in Pharo, communication via FFI)

The Performer subclasses implement the `play` method

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Inside a Performer

```
Performer >>playFor: aNumberOfSteps

            performance bpm: 60 / (performance freq * 4).
            performance transportStep: 0.
            performance activeProcess: ([aNumberOfSteps timesRepeat:
                [(Delay forSeconds: performance freq) wait.

            "sequencers scan"

            performance valuesDo: [ :seq |

               (seq gates wrap: performance transportStep) = 1 ifTrue:
                  [ self playEventAt: seq noteIndex in: seq.

               "increment note Index"

                  seq noteIndex: seq noteIndex + 1 ] ] ] forkAt: Processor
timingPriority - 2.

               "step is incremented anyway"
                  performance incrementTransportStep ] ] forkAt:
                  Processor timingPriority - 1)
```

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo Association

Play events(phausto)

```
PerformerKyma >> playEventAt: anIndex in: aSequencer

        self subclassResponsibility
```

```
PerformerPhausto >> playEventAt: anIndex dsp: aDsp freq: aFrequency in: aSequencer

        | dur aParameterList |
        dur := aSequencer durations asDirtArray wrap: anIndex.
        aParameterList := self performance activeDSP allParameters.

        aSequencer extraParams keysAndValuesDo: [ :k :v |
                        aParameterList
                                setValue: (v wrap: anIndex)
                                parameter: aSequencer seqKey , k asString
                                forDsp: aDsp ].

        aParameterList
                setValue: (aSequencer notes wrap: anIndex) midiNoteToFreq
                parameter: aSequencer phaustoNoteDestination
                forDsp: aDsp.
        aParameterList
                trig: aSequencer phaustoGateDestination
                for: dur * aFrequency * (aSequencer gateTimes wrap: anIndex)
                forDsp: aDsp
```

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Play events(MIDI)

```
PerformerKyma >> playEventAt: anIndex in: aSequencer

        self subclassResponsibility
```

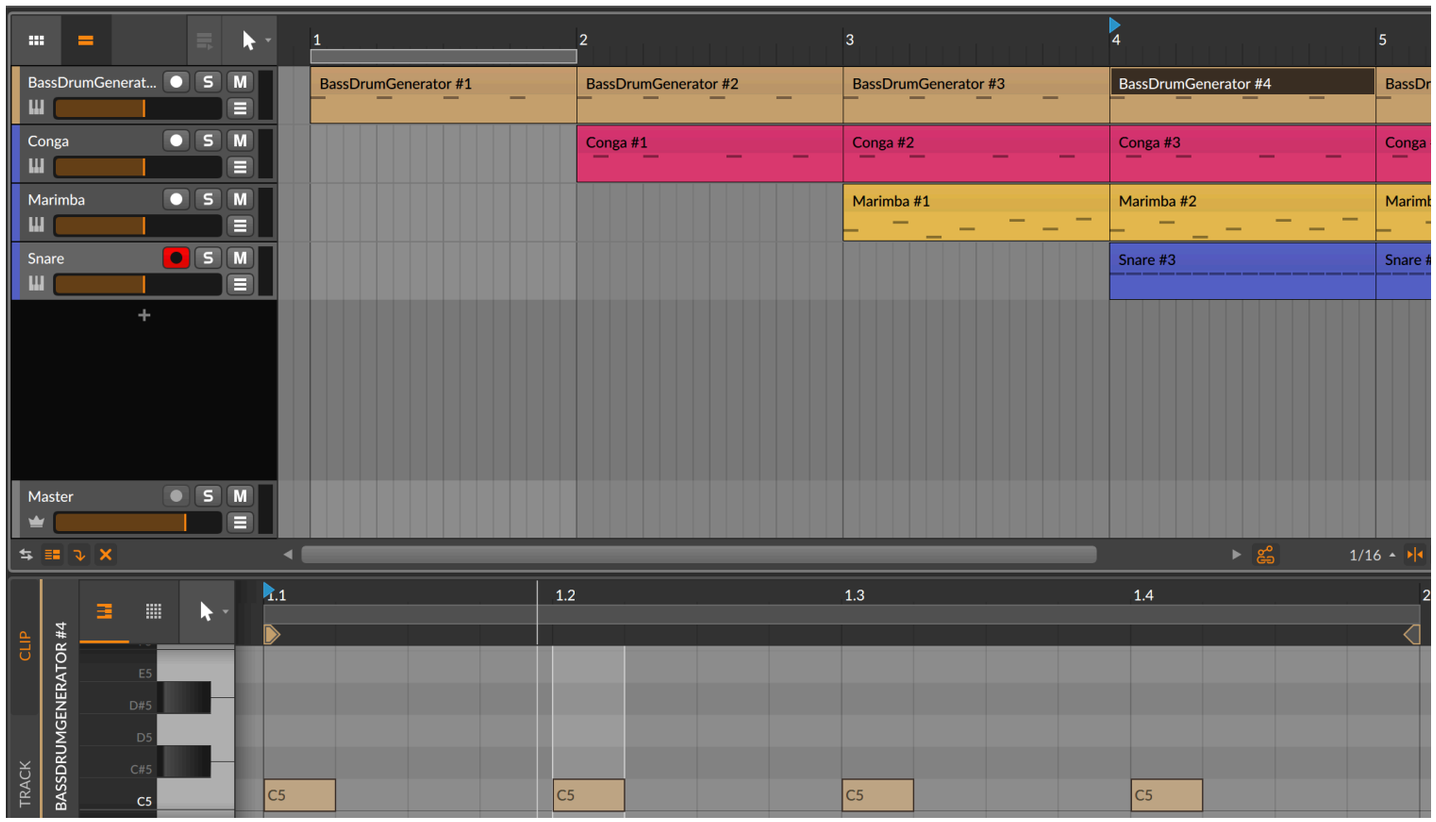
```
PerformerMIDI >> playEventAt: anIndex in: aSequencer

        | gateTime dur midiNote mch stepDuration midiSender |
        freq := Performance uniqueInstance freq.
        gateTime := 0.9. "must be changeable"
        midiSender := PerformerMIDI midiOut.
        mch := aSequencer midiChannel.
        stepDuration := Performance uniqueInstance freq.
        midiNote := aSequencer notes asDirtArray wrap: anIndex.
        dur := aSequencer durations asDirtArray wrap: anIndex.
        midiSender
                playNote: midiNote
                onChannel: mch
                duration: dur * freq * gateTime.
```

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Many sequencers

Virtually has many as you need

A Sequencer resemble a *track* in a Digital Audio WorkStation



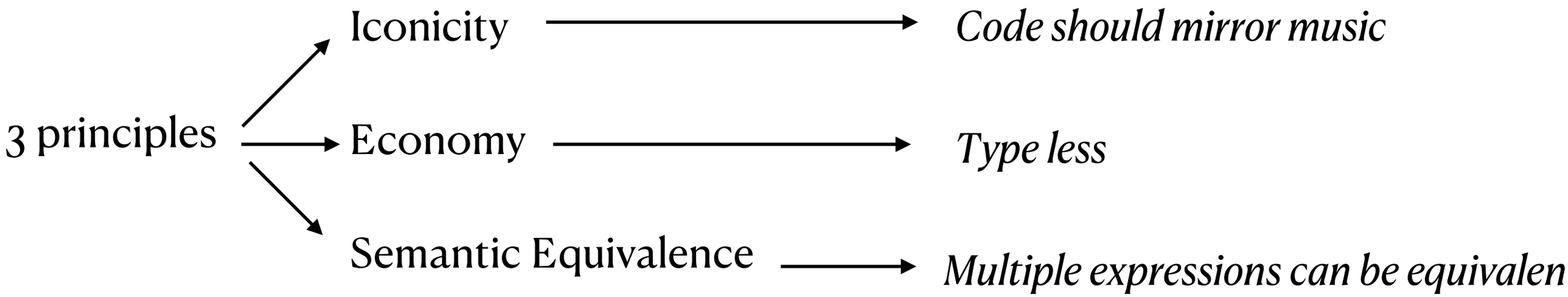
But in Coypu its values can have different size !

```
16 semiquavers index: '1 , 7' ; notes: '62 , 65 , 67' to: #speakspell
```

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo Association

A NEW MUSIC DIALECT



Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo ASSOCIATION

A NEW MUSIC DIALECT

Create `Sequencers`

- From Arrays → `#(1 0 0 1 0 1 0 0 1 1 ) asSeq.`
- With random generators → `16 randomTrigsWithProbability: 67.`
- From euclidean rhythms → `#(3 16) euclidean.`
- With a *string notation* → `'40*2 , 57 , ~ , 65/3 , ~*5' asDirtNotes.`
- From a *rhythm* name → `16 cumbiaClave.`

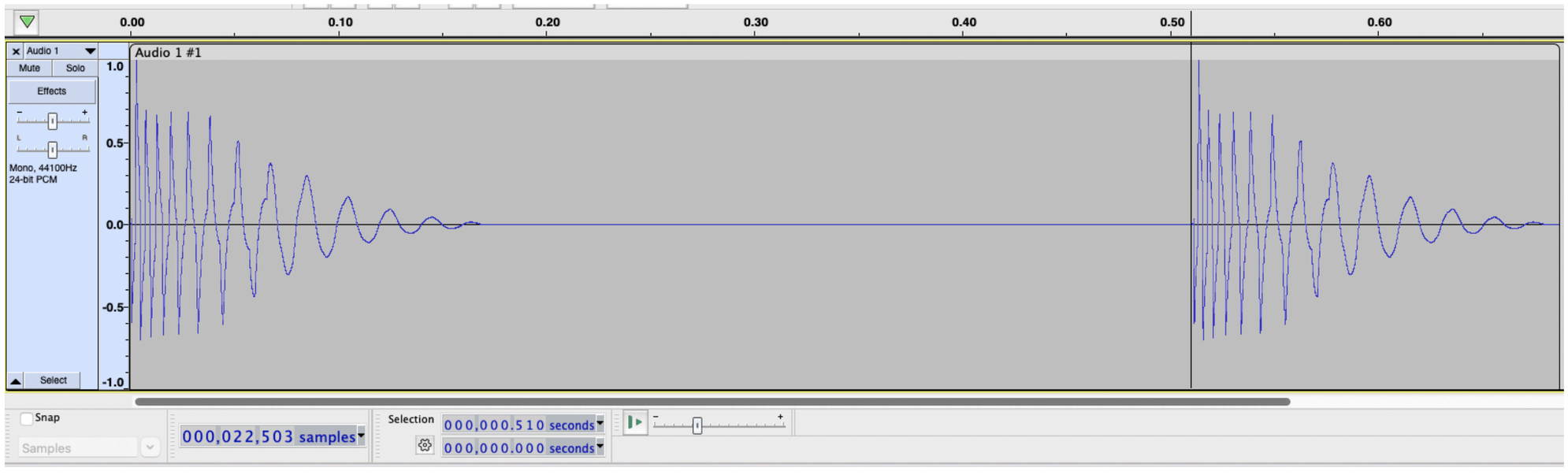
Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo Association

Road to fame

We consider Coypu to still be in its beta phase. In order to proceed toward a stable release, several current limitations need to be addressed:

1. No current support for cyclical structures (e.g., TidalCycles-style patterns).
2. Timing resolution and rhythmic subdivision control currently unavailable.
3. The average jitter in the performance playhead advancement is approximately 1 ms.



We're working toward a solution for testing performance in dynamic, on-the-fly music systems like Coypu.

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo Association

Conclusions

ICLC2025 (Algo:noises at Laut, Barcelona)

Festival della Robotica 2025 (Workshop, Pisa)

IFC2024 (Torino)

ESUG2024 (Lille)

ICLC2024 (Algorave at System, Shanghai)

Algorave at OHM (Berlin , 2024)

ESUG2023 (Lyon)

ICLC2023 (Algorave + Workshop, Düsseldorf)

ESUG2023 (Novi Sad)

Cyberspeak (Milano, 2022)

Still in beta,
but already in the wild

Empirical investigation into teaching methods and their real-world application is essential for effective evaluation.

We need more workshops and more users,
to spread everywhere.

Domenico Cipriani, Sebastian Jordan Montaño, Nahuel Palumbo, Stéphane Ducasse

Pharo ASSOCIATION