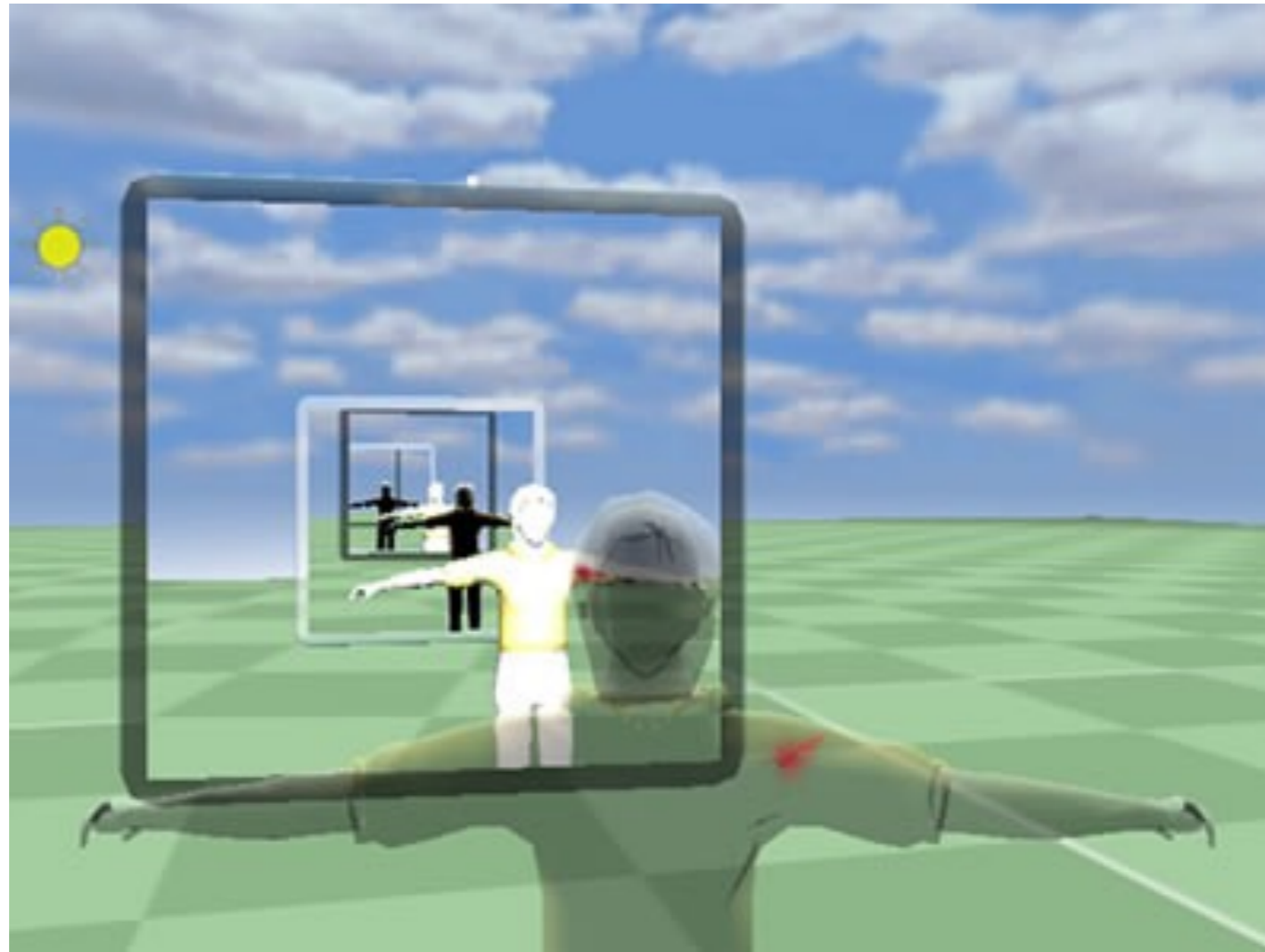


Towards the Web for Virtual Being



Screenshot from Krestianstvo SDK 1.0

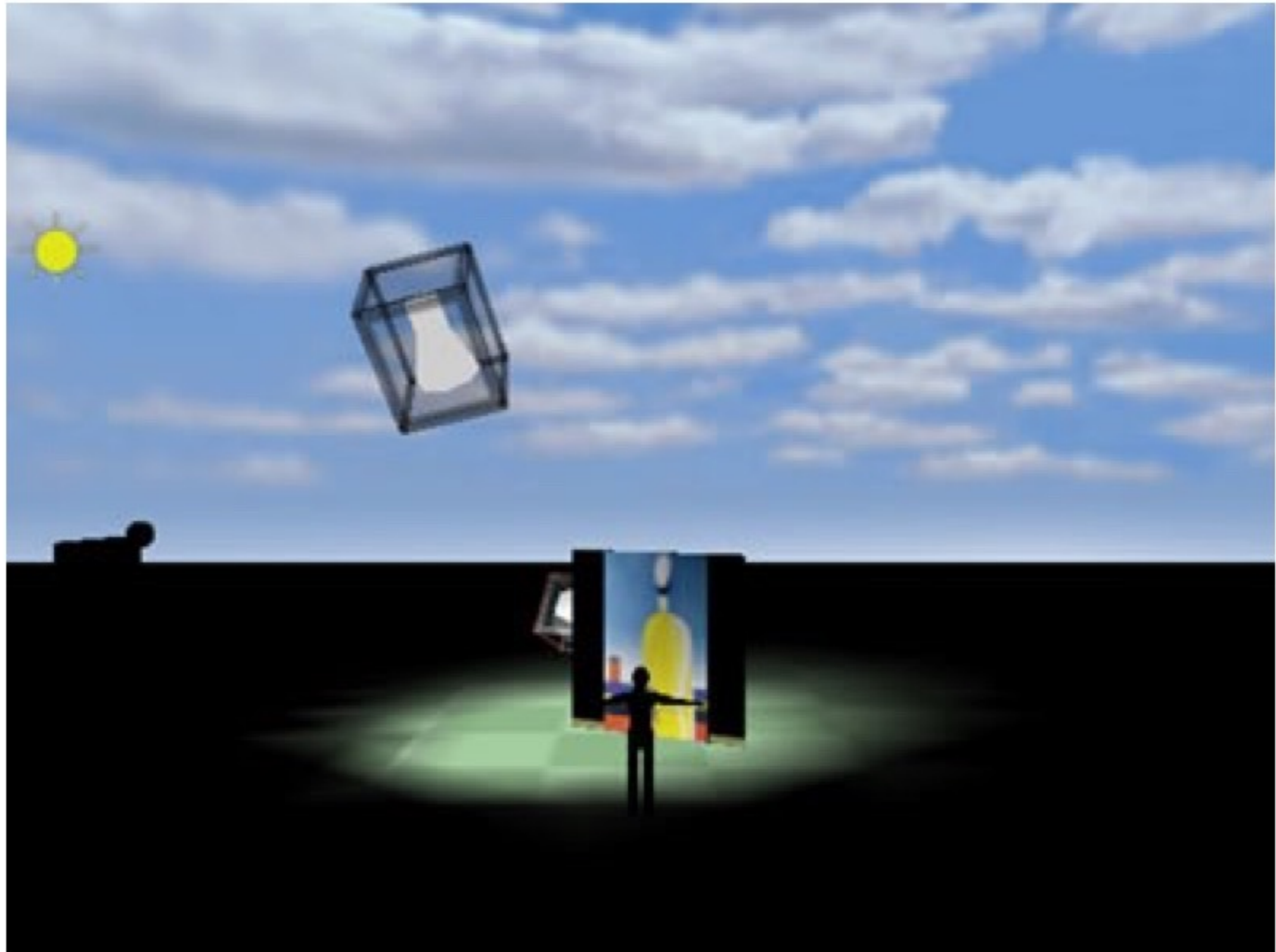
Nikolai Suslov

Fund for Supporting Development of RT, Russia

SuslovNV@krestianstvo.org

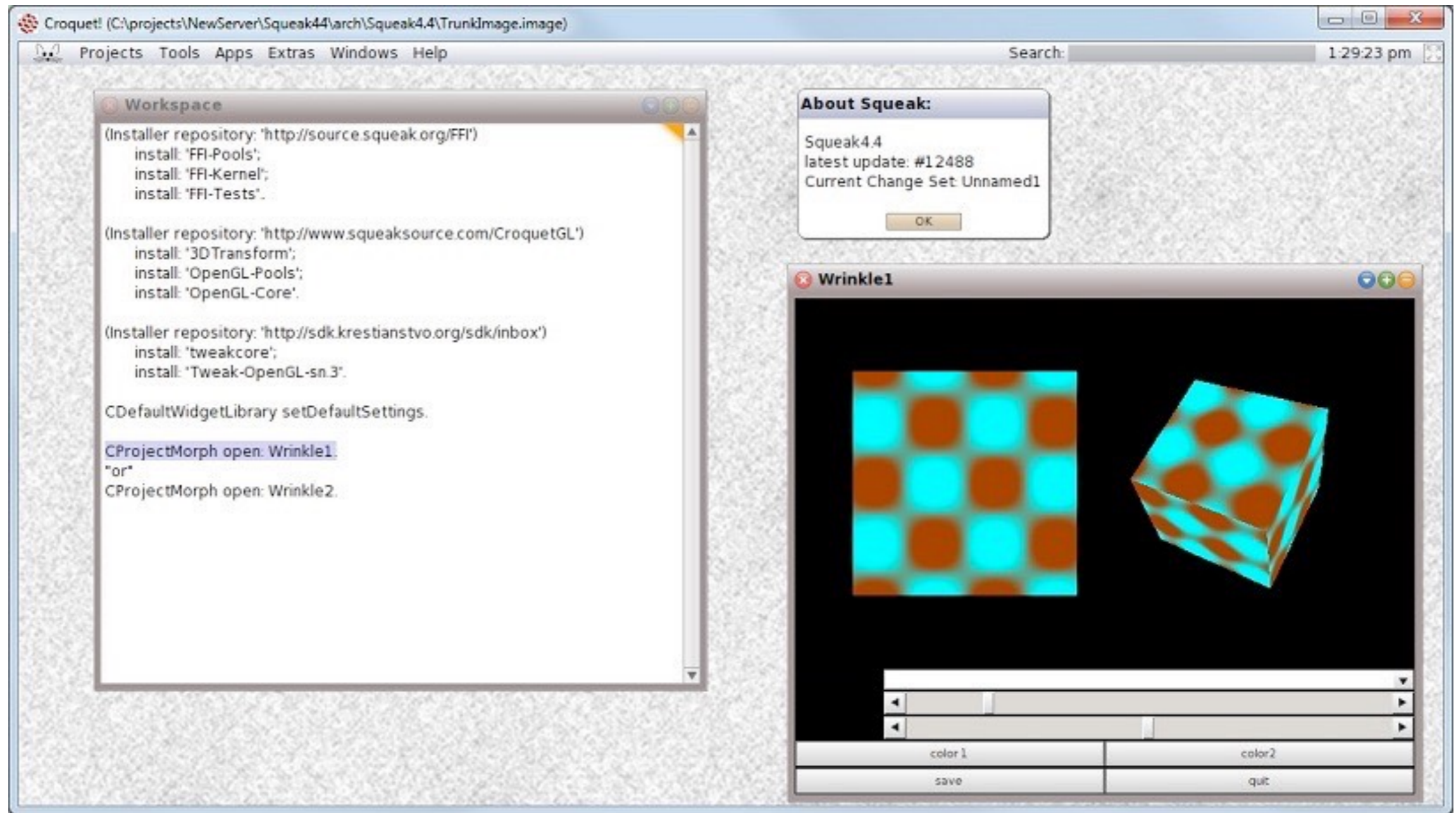
<https://www.krestianstvo.org>

State of the Art



Screenshot from Krestianstvo SDK 1.0

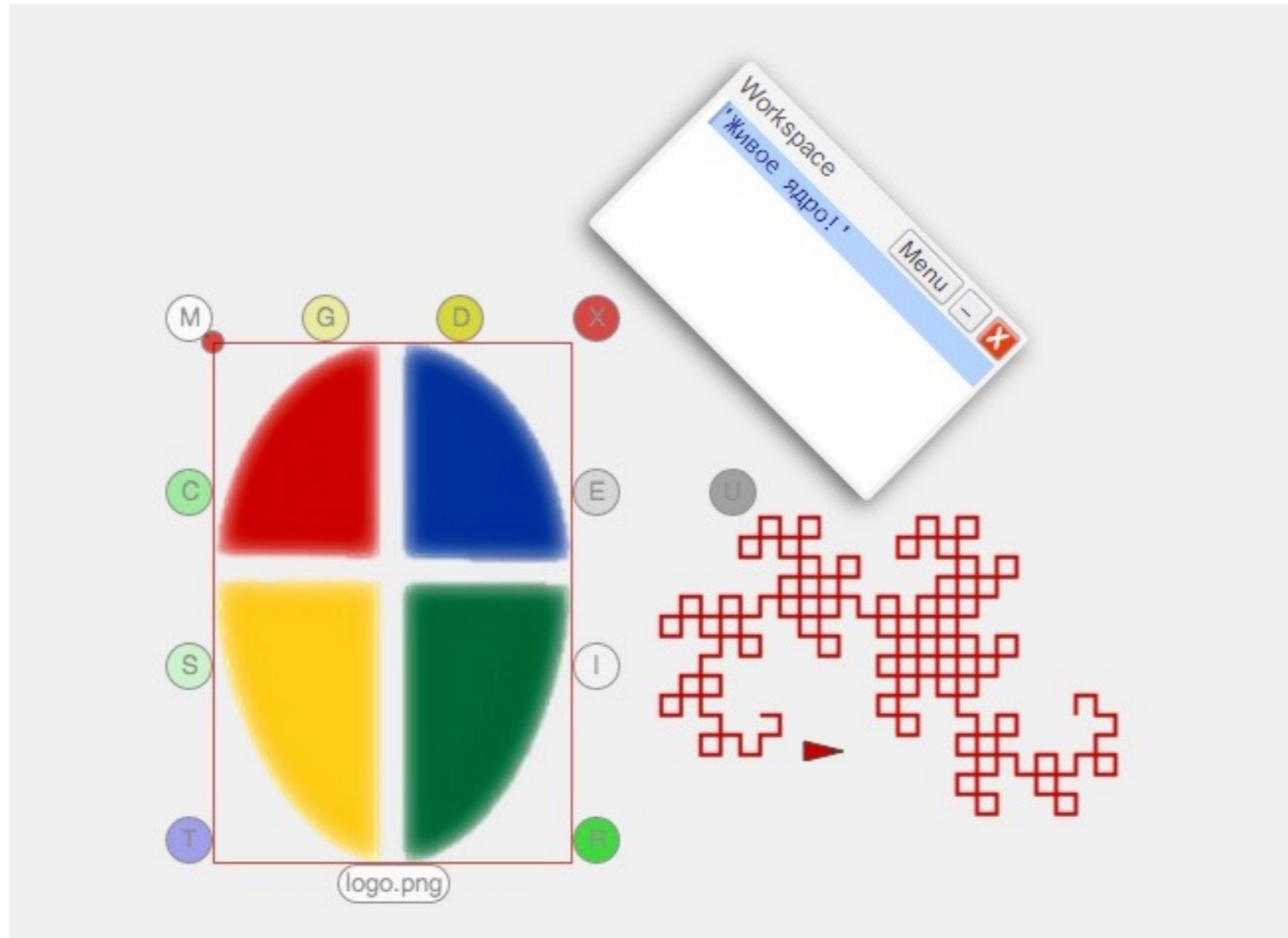
SELF EXPLORATIVE ENVIRONMENTS



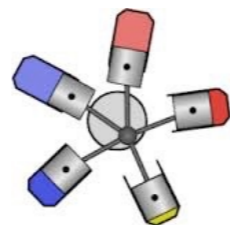
Screenshot from Squeak Smalltalk with OpenGL



SELF EXPLORATIVE ENVIRONMENTS in browser



Screenshot from Lively Kernel



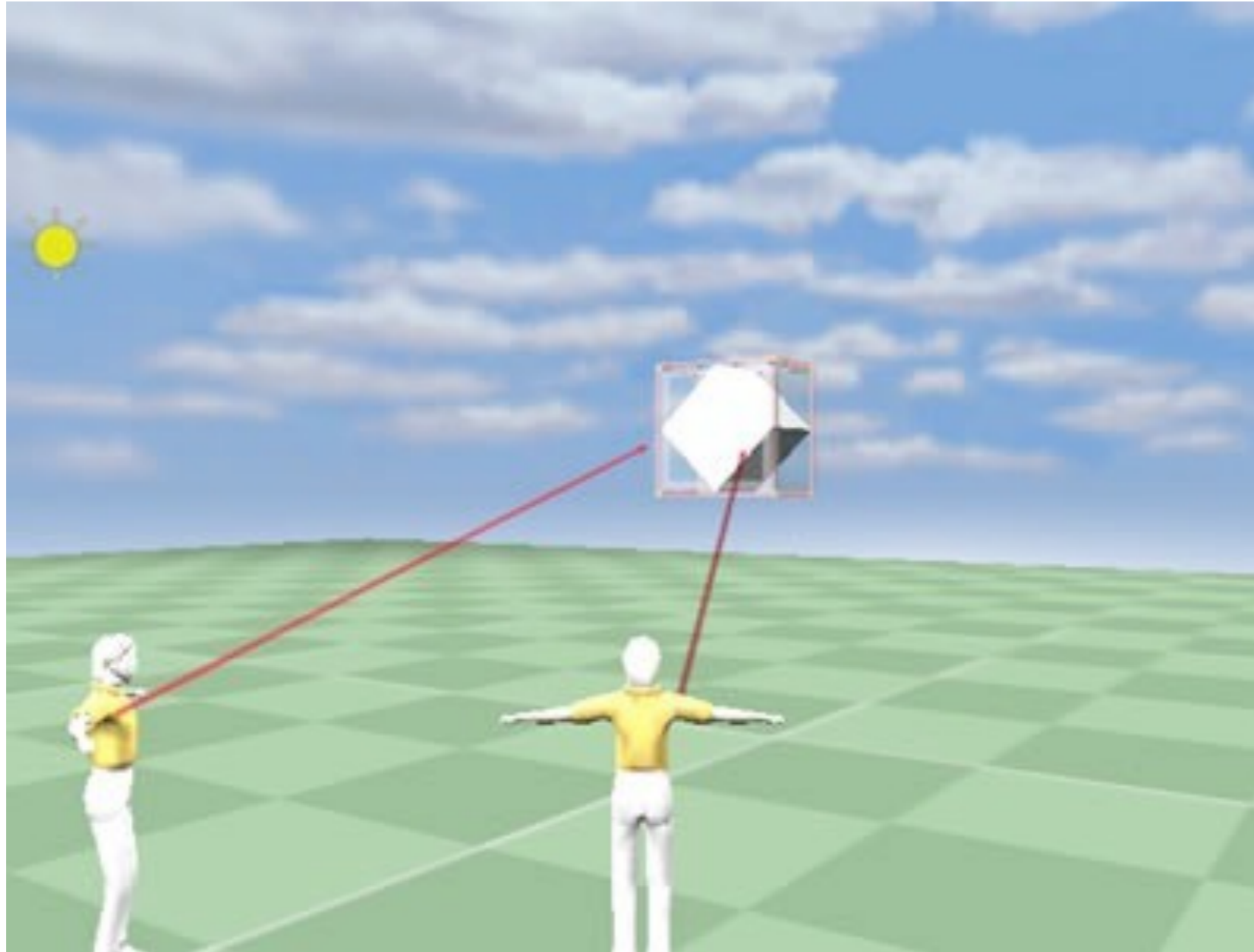
Lively Kernel

Virtual Worlds

- Immersive Terf (<http://www.3dicc.com>)
- HighFidelity (<https://highfidelity.io>)
- AltspaceVR (<https://altvr.com>)
- Sansar (<https://www.sansar.com>)
- SpatialOS (<https://spatialos.improbable.io>)

*Still, these are all desktop apps, they are considered as Web ready.

COLLABORATIVE SELF EXPLORATIVE ENVIRONMENTS

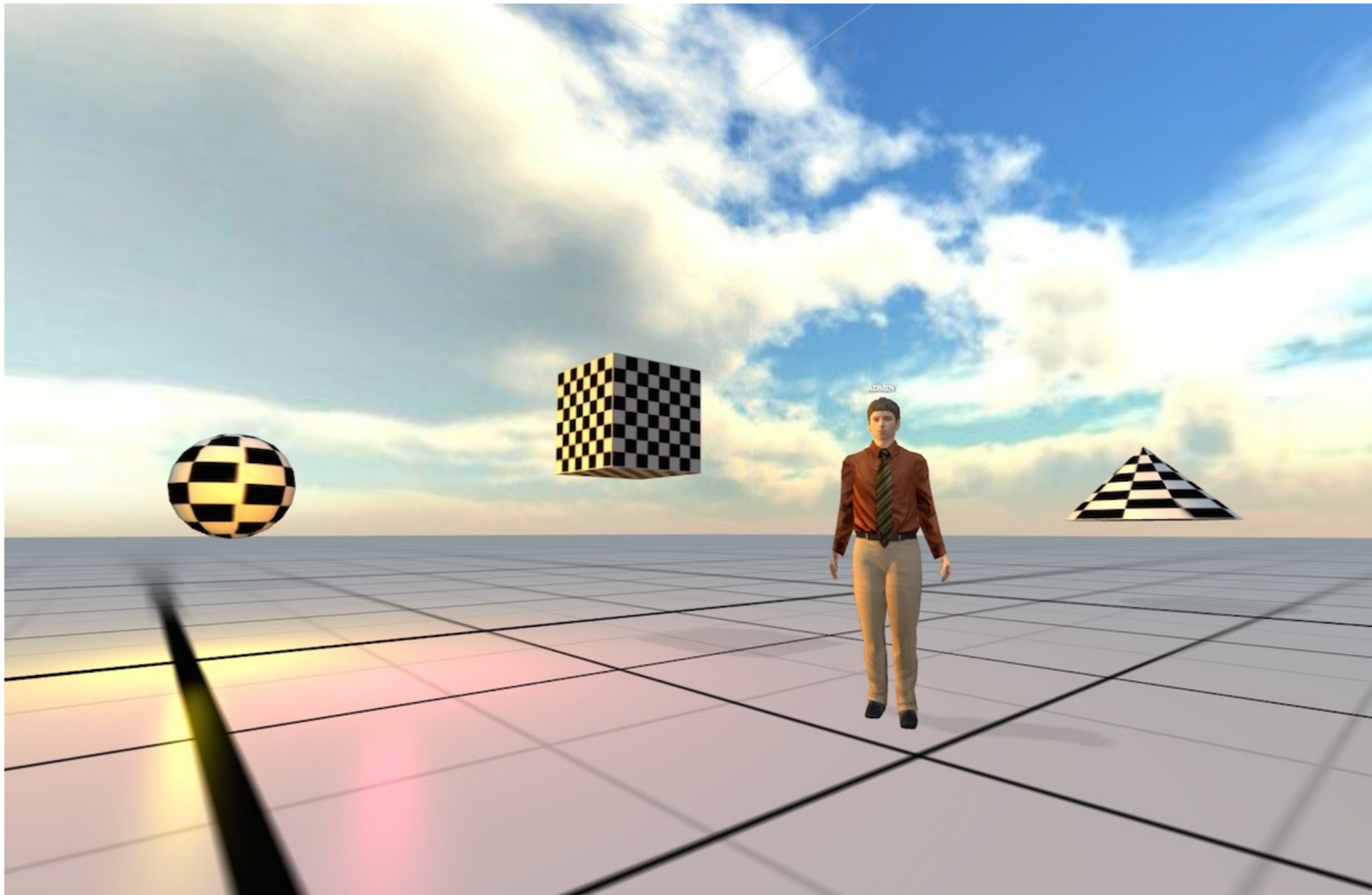


Screenshot from Krestianstvo SDK 1.0



COLLABORATIVE SELF EXPLORATIVE ENVIRONMENTS

in browser

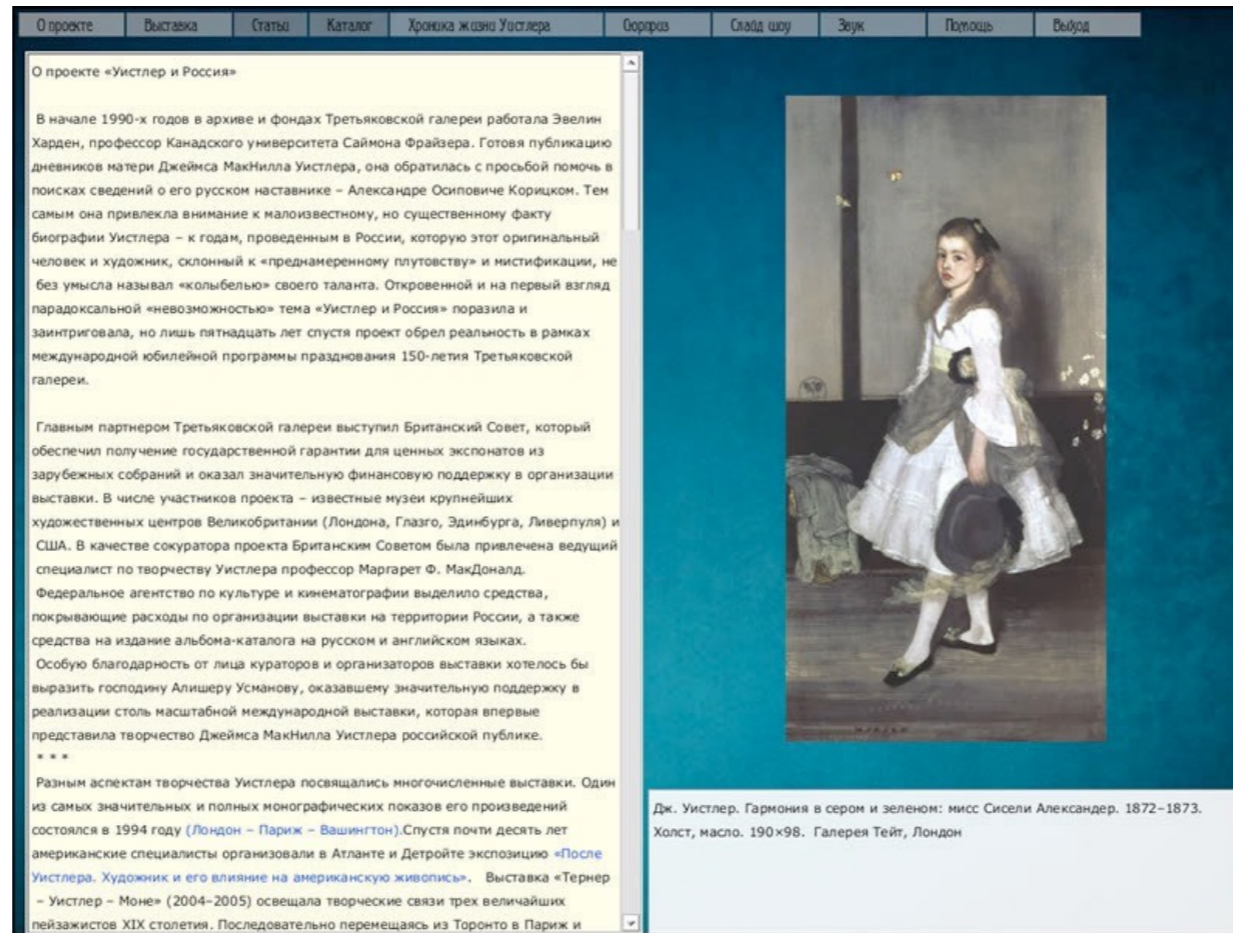


Screenshot from ADL Sandbox & Krestianstvo VLE



Virtual World Framework

Sophie - a project of The Institute for the Future of the Book

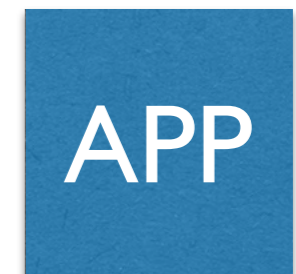
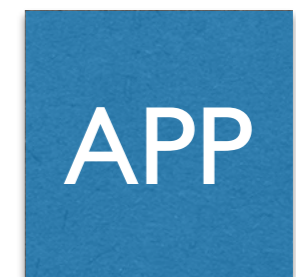
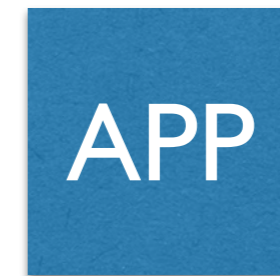
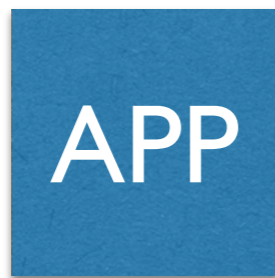


Screenshot from multimedia disc, based on Sophie

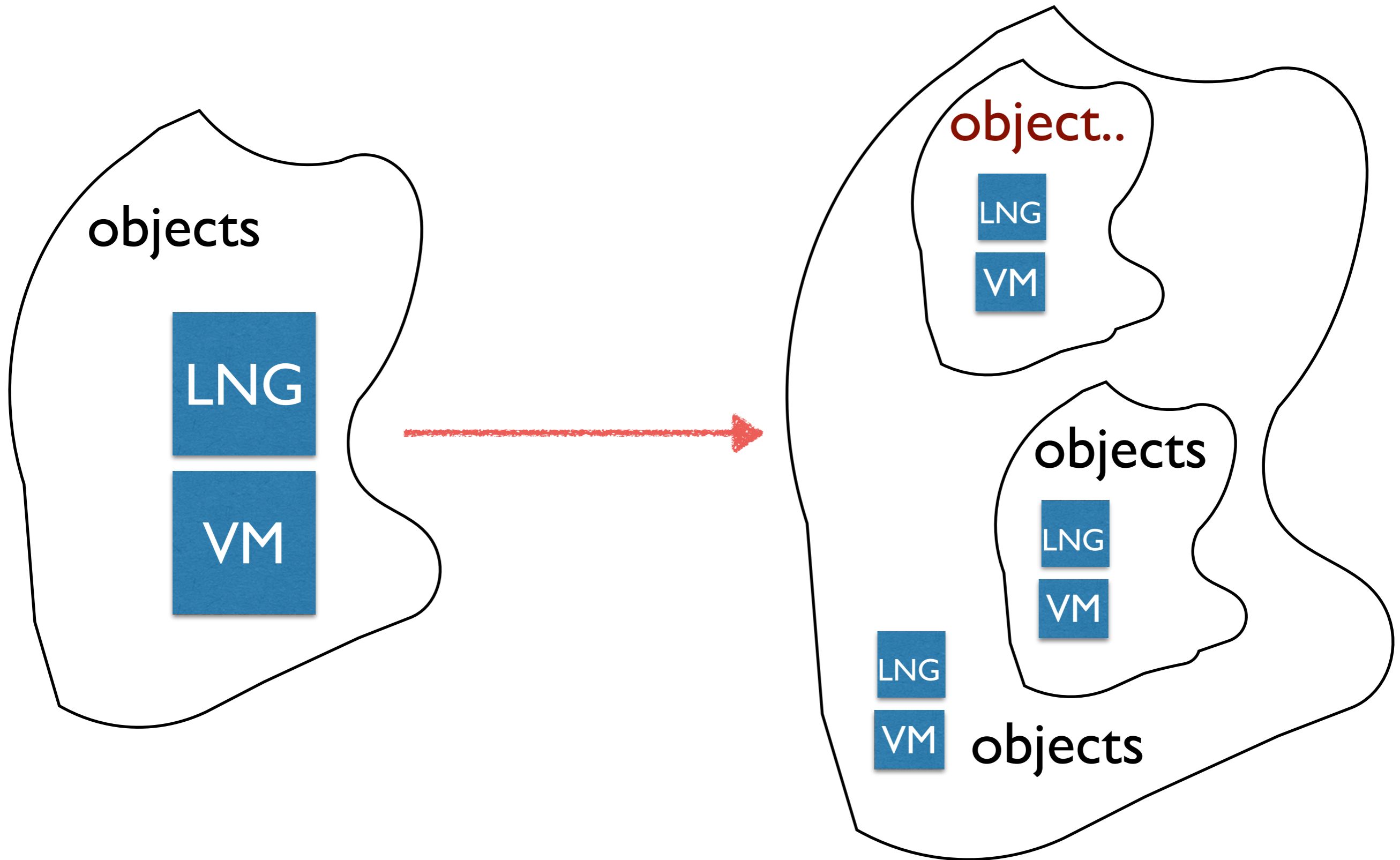
.media

.media

.media



OMeta / Ohm - object-oriented language for pattern matching



Software agents by Henry Lieberman



Virtual Time in Open Croquet architecture

the new model of distributed computation

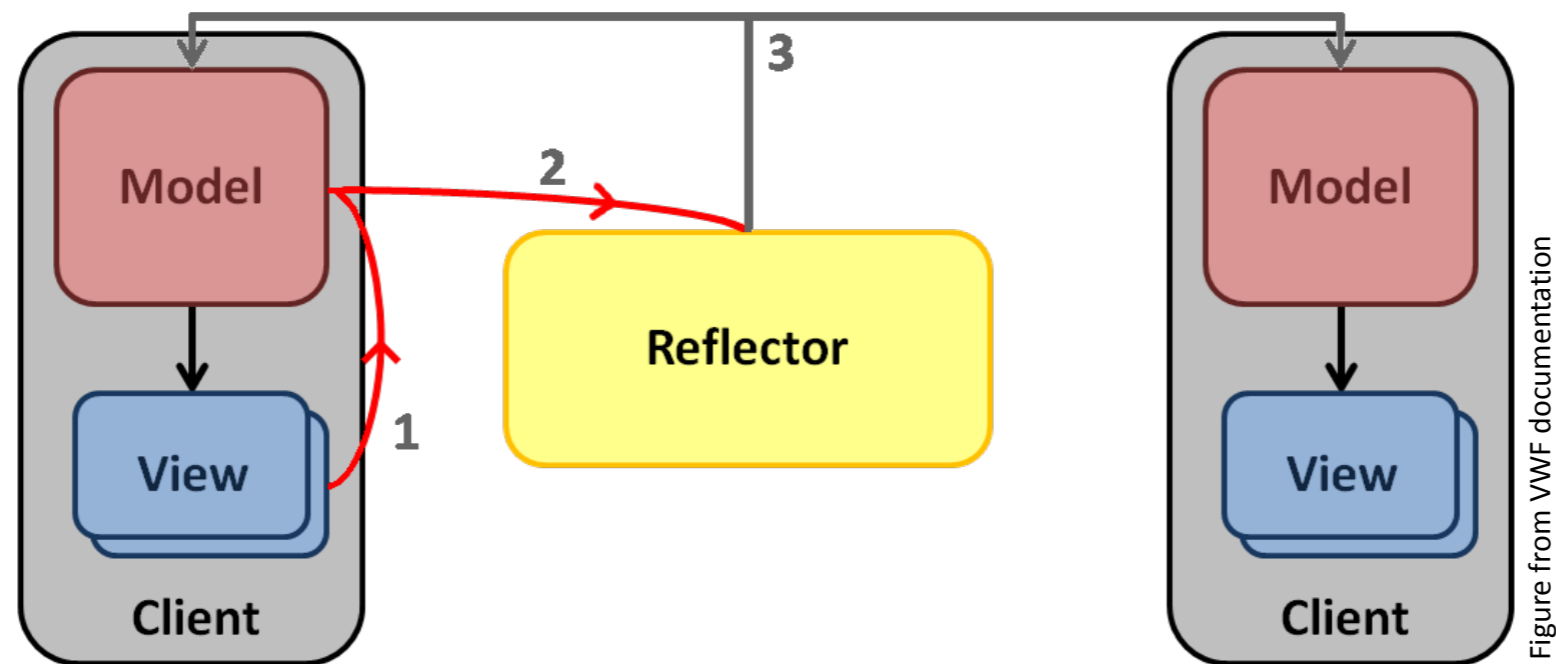
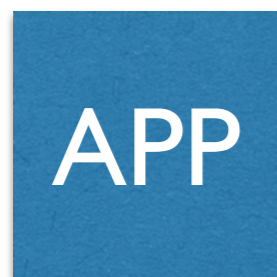


Figure from VWF documentation

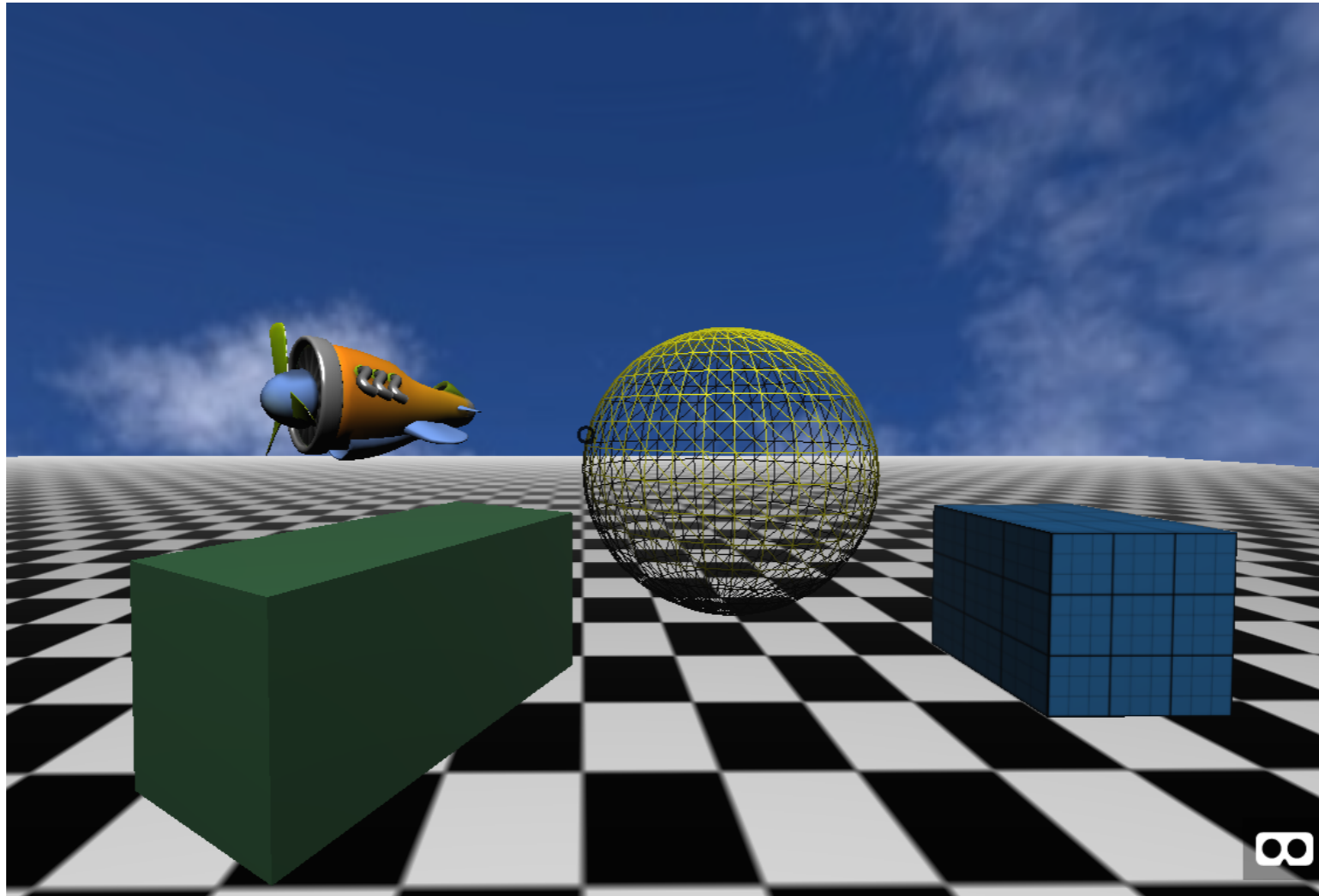
synchronisation



Virtual Time



A-Frame - declarative components for 3D graphics, interaction and **Web VR** in Web browser (aka new **VRML**)



Screenshot from livecoding.space demo app

```
<a-scene>
```

```
  <a-box position="-1 0.5 -3" rotation="0 45 0" color="#4CC3D9"></a-box>
```

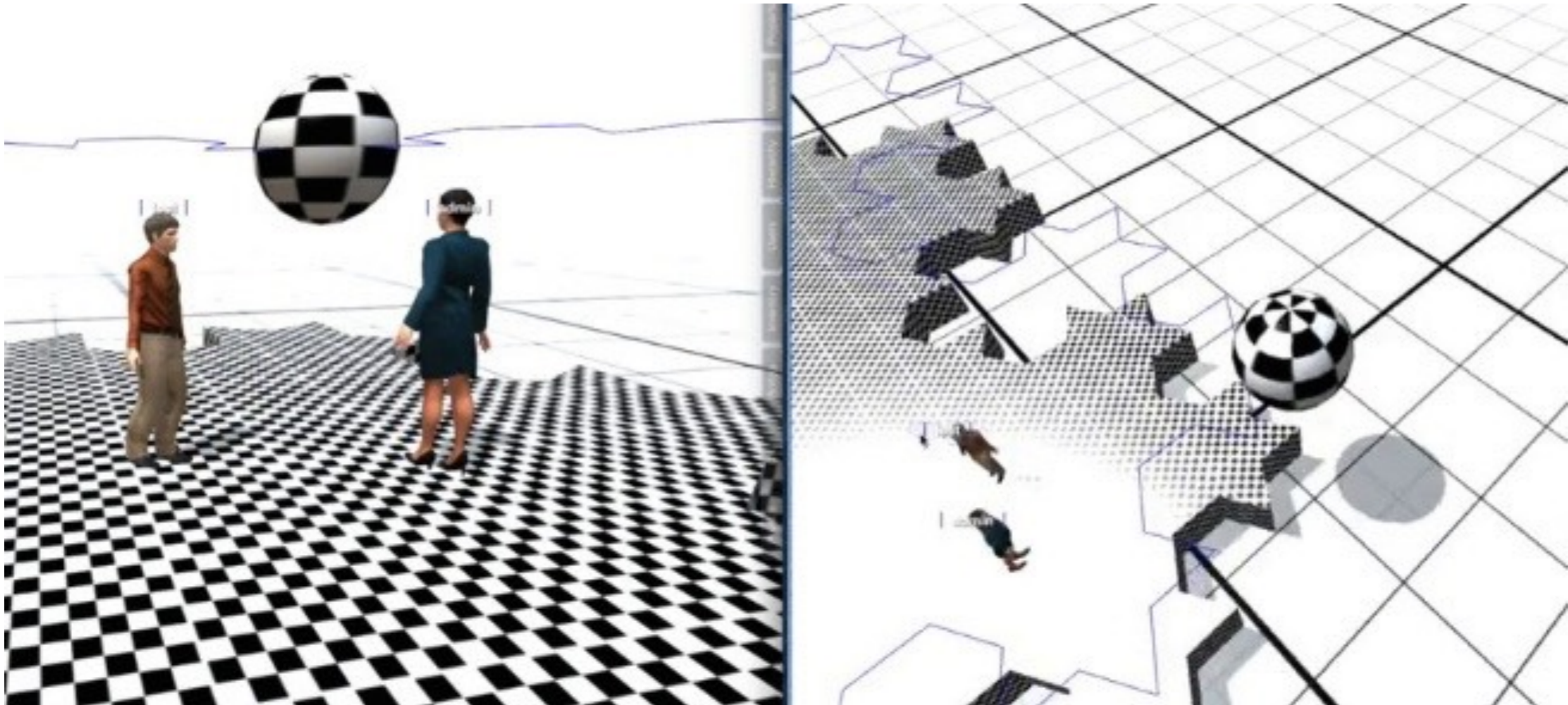
```
  <a-sphere position="0 1.25 -5" radius="1.25" color="#EF2D5E"></a-sphere>
```

```
  <a-plane position="0 0 -4" width="4" height="4" color="#7BC8A4"></a-plane>
```

```
  <a-sky color="#ECECEC"></a-sky>
```

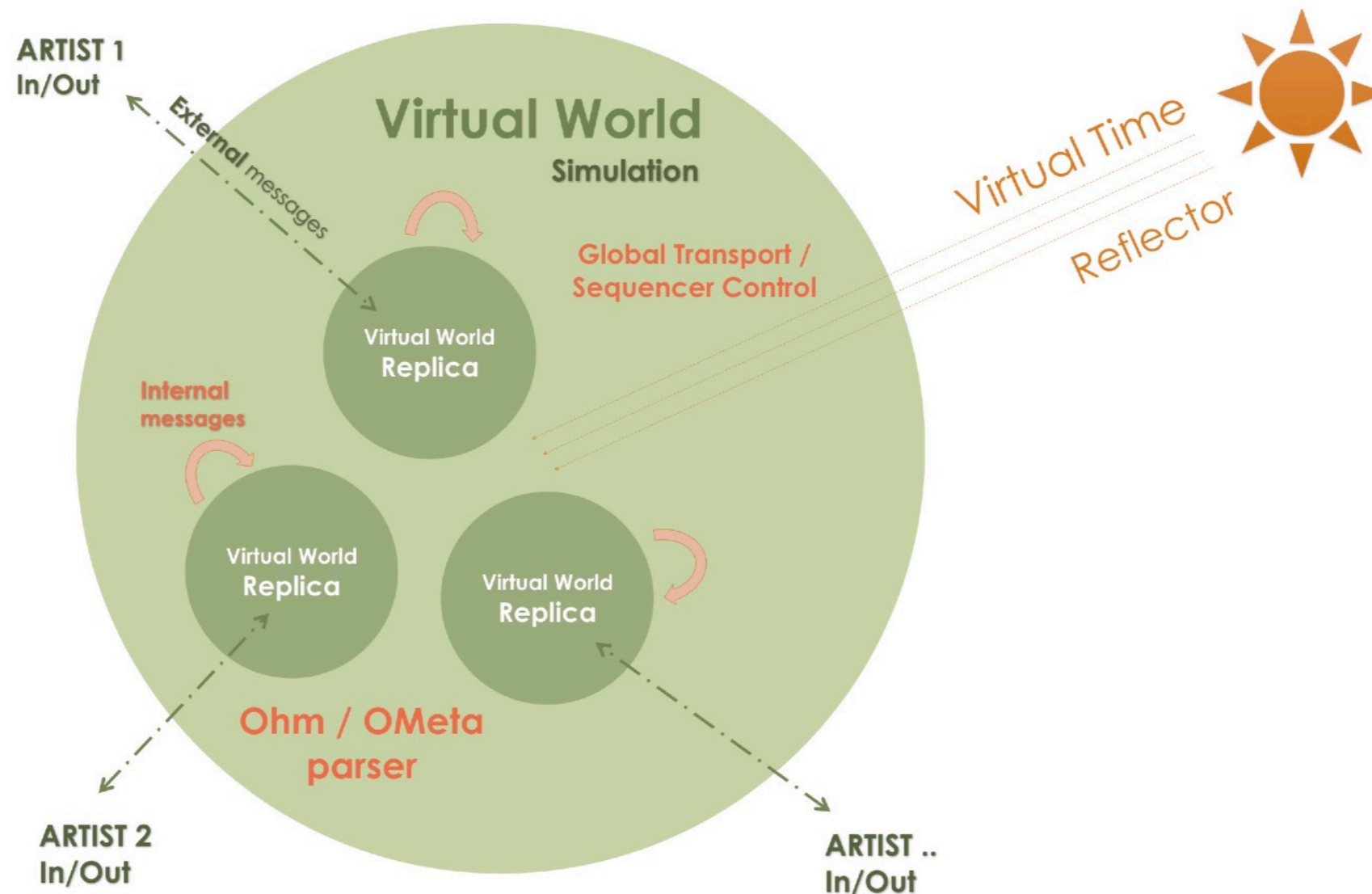
```
</a-scene>
```

The prototype of **VIRTUAL BEING**



Screenshot from ADL Sandbox & OMeta integration prototype

Integration of OMeta/Ohm into Virtual World Framework



OSC /messages
JSON objects
Serial data

Software	Sensors
SuperCollider	LeapMotion
Max/MSP	Kinect
Pure Data	Wii
...	...

Ohm's model/view driver for Virtual World Framework

```
settingProperty: function( nodeID, propertyName, propertyValue ) {  
...  
    node.lang.source = propertyValue;  
    node.lang.grammar = ohm.grammar(propertyValue);  
    node.lang.semantics = node.lang.grammar.createSemantics();  
}
```

App configuration

```
---  
info:  
  title: "Ohm calculator Example App"  
model:  
  vwf/model/ohm:  
  vwf/model/aframe:  
view:  
  vwf/view/aframe:  
  vwf/view/ohm:  
  vwf/view/editor-live:
```

A-Frame components inside VWF with Ohm grammars

calcText:

```
extends: http://vwf.example.com/aframe/atext.vwf
```

```
properties:
```

```
  value: "1 * pi"
```

```
  color: "#b74217"
```

```
  position: [-1, 2.5, -2]
```

```
  scale: [2, 2, 2]
```

```
children:
```

```
  calcLang:
```

```
    extends: http://vwf.example.com/ohm/node.vwf
```

```
    properties:
```

```
      grammar:
```

```
      semantics:
```

```
      ohmLang: |
```

```
        Arithmetic {
```

```
          Exp
```

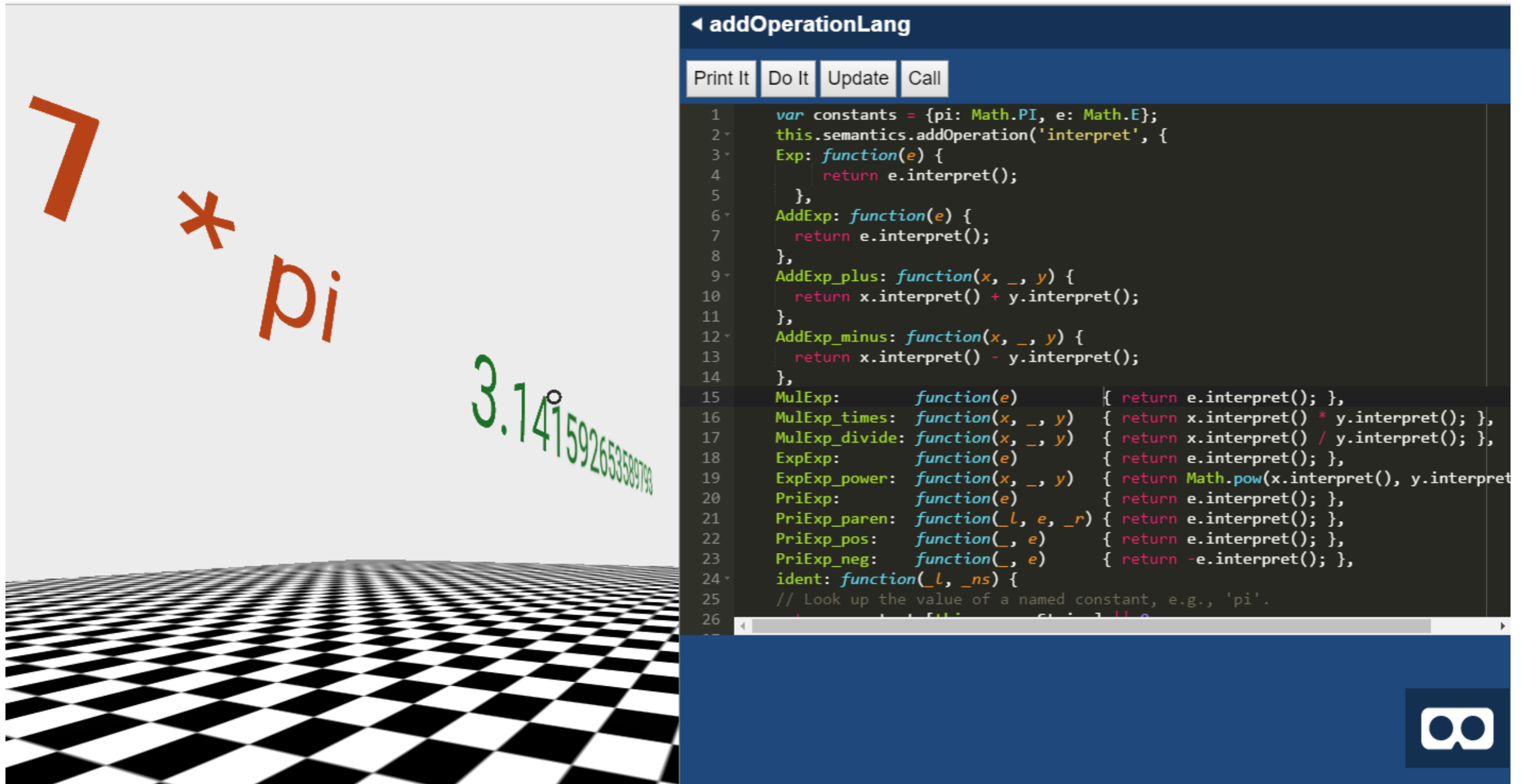
```
            = AddExp
```

```
          AddExp
```

```
            = AddExp "+" MulExp -- plus
```

```
          ...
```

Live Coding editor features for VWF components on a web page



The screenshot shows a live coding editor titled "addOperationLang". It features a dark background with syntax-highlighted JavaScript code. At the top of the editor, there are four buttons: "Print It", "Do It", "Update", and "Call". The code defines a set of constants and operations for a virtual world framework (VWF) component. The code includes functions for adding operations, handling expressions, and performing arithmetic operations like addition, subtraction, multiplication, and division. It also includes functions for handling mathematical constants like pi and e, and for parsing expressions with parentheses and unary operators.

```
1 var constants = {pi: Math.PI, e: Math.E};
2 this.semantics.addOperation('interpret', {
3   Exp: function(e) {
4     return e.interpret();
5   },
6   AddExp: function(e) {
7     return e.interpret();
8   },
9   AddExp_plus: function(x, _, y) {
10    return x.interpret() + y.interpret();
11  },
12  AddExp_minus: function(x, _, y) {
13    return x.interpret() - y.interpret();
14  },
15  MulExp: function(e) { return e.interpret(); },
16  MulExp_times: function(x, _, y) { return x.interpret() * y.interpret(); },
17  MulExp_divide: function(x, _, y) { return x.interpret() / y.interpret(); },
18  ExpExp: function(e) { return e.interpret(); },
19  ExpExp_power: function(x, _, y) { return Math.pow(x.interpret(), y.interpret()); },
20  PriExp: function(e) { return e.interpret(); },
21  PriExp_paren: function(_l, e, _r) { return e.interpret(); },
22  PriExp_pos: function(_, e) { return e.interpret(); },
23  PriExp_neg: function(_, e) { return -e.interpret(); },
24  ident: function(_l, _ns) {
25    // Look up the value of a named constant, e.g., 'pi'.
26  }
```

Screenshot from livecoding.space demo app

DEMO

<https://livecoding.space>

Statement for discussion:

Moving from Web pages to Web apps as Desktop ones could be a “falling into the same trap”. **Virtual Worlds** could help to avoid this and get the Web back to the future of powerful ideas from Self language and Amiga OS.