Situated Interaction

Wendy Mackay & Michel Beaudouin-Lafon
28 January 2021

mackay@lri.fr   *mbl@lri. fr*

This week:

Discuss exercises: #8, #9

Disscussion: Framework

Guest Lectures:

    Fanis Tsandilas   Structgraphics
    Viktor Gustaffson   Narrative Substrates
    Janin Koch   Semantic Collage & Image Sense

Exercise #8   Finding Structures

Group exercise

1. Pick three activities.
2. Identify at least five structured ways to organize information for that activity.
3. Identify the type of computer structure and what it is for.

Example: *Activity:*   *Cooking*
Shopping list   *paired list*   *specify amounts of ingredients*

Exercise #9　　Generative Design

Group exercise:
Brainstorm how to use: *reification, polymorphism & reuse*
to manipulate *graphical styles* in a graphical editor

1. Think of a more powerful instrument for defining
   and/or modifying graphical styles

2. Explain how the idea embodies the principles

3. What is the substrate?

4. How is it discoverable? appropriable? expressive?

Framework: Instruments and Substrates

	Instruments	Substrates
	Actions (input)	Effects (output)
Reification	If command disappears, make it persist	If a relationship disappears, make it persist
Polymorphism	Make it handle multiple types of objects	Make it handle multiple types of relationships
Reuse	Make it reuse user input	Make it reuse user output

StickyLines example

	Instruments	Substrates
	Actions (input)	*Effects (output)*
Reification	If command disappears, make it persist • *Make alignment tool, apply to groups of objects*	If a relationship disappears, make it persist • *Alignment relationship persists if objects move*
Polymorphism	Make it handle multiple types of objects • *Can align shapes, text, windows, etc.*	Make it handle multiple types of relationships • *Can align and distribute objects*
Reuse	Make it reuse user input • *Remember and reuse previous object alignments*	Make it reuse user output • *Remember and resuse previous relationships*

New example?

	Instruments	Substrates
	Actions (input)	*Effects (output)*
Reification	If command disappears, make it persist	If a relationship disappears, make it persist
Polymorphism	Make it handle multiple types of objects	Make it handle multiple types of relationships
Reuse	Make it reuse user input	Make it reuse user output