Coding scheme companion guide for the categories of (1) Sketches, (2) Focus fixations and transitions, and (3) Reasoning activities

Original coding June, 2012 by Nicolas Mangano, Thomas LaToza
Last updated September 4th, 2014 by Nicolas Mangano

Overview

This companion guide is intended to extend the explanations for each category given in the paper titled, “How Software Designers Interact with Sketches at the Whiteboard”. This guide provides additional detail of the practical aspects of how we coded each category, providing examples and key phrases intended to help increase consistency when using this coding scheme. Also, this is not a perfect document by any means, but rather is a collection of notes that the coders of the videos used to help them reach an agreement in how to code each category.

In this coding scheme, we cover the following parts:

- creating a sketch reference sheet before watching the video
- coding the sketches they create
- coding the transitions made between videos
- coding particular activities that happen (which may overlap)

Since this is quite a bit to code all at once, I first recommend creating the sketch reference sheet. After that has been created, I recommend watching the video in increments of 5 minutes, coding the following items separately:

- first code the transitions, ignoring the sketchess and activity, since tracking attention is likely to be the most difficult task while coding
- review the 5 minute increment, this time coding the sketch and activity categories.

Video coding preparation

Before performing any of the coding schemes below, please first prepare a sketch reference document for each video. Preparing this document ahead of time will make the coding schemes (particularly Category 1) easier to perform.

In order to code the video, perform the following steps

1. Open video in VLC

2. Take screenshots of video every 5 minutes
    a. Ctrl + T jumps to a video position on windows / Ctrl + J on mac.

   b. First position will be 0m, then 5m, then 10m, etc. (VLC only allows jumping by seconds so instead: 0s, then 300s, then 600s, etc.)
   c. Continue until video ends, then repeat on next segment if video continues on another file.

3. Open screenshots one at a time, in the order of chronology, and take a screenshot of each new "sketch cluster". Save sketches using the following pattern: sketch1.png, Sketch2.png, etc. The numbering of new sketches in each screenshot begins from left to right, top to bottom. Note: this may not reflect the absolute chronology of sketches, instead the goal is to maintain consistency in sketch labeling between coders (without the need to watch the videos first).

4. Create a new document and paste all images into this document. For each sketch, note the following information:

   Sketch number:
   Sketch type:
   Visual syntactic elements:

   Example:

   Sketch number: Sketch 1
   Sketch type: Map
   Visual syntactic elements:
   - Map grid lines
   - labels for map grid lines
   - Compass annotation
   - Intersection focus point (box drawn around an intersection)

   The visual syntactic elements (VSE for short) do not need to be a comprehensive list, this is only made to prime yourself before watching the video. While watching the video, this list is likely to grow.

Category 1 Codings - Kinds of sketches they create

Coding format
(time): <Sketch number><Sketch domain><Sketch type><VSE introduced>
e.g., (0:07:49.9): <1><user interface><map><1>

(time): <Sketch number><Sketch domain><Sketch type><erased>
e.g., (0:07:49.9): <1><user interface><map><erased>

Sketch domains (these only belong to the instant this is record, and can change over time)
* <user interface>
* <system>
* <domain>
* <requirements>

Description
If the designers steal a formal convention (VSE) from another notation, then, in the coding, note that by preceding <VSE introduced> with the type. E.g., *(0:07:49.9): <1><user interface><map><class sketch-5>*

In this category, we are recording all instances in which a new VSE is introduced, as well as when it is erased. When new sketches are created, several VSEs may be introduced rapid in rapid succession. In these cases, the coder should record each of these as distinct event with separate time codings.

Please refer to document, *"Visual Syntactic Elements Reference Document"*, for a full list of available sketch and formality types. If the identified sketch or formality type is not included in the list, the coder must add a new entry and number.

Category 2 Codings - How they move between sketches

Coding format
<target sketch><source sketch> {<Reason for shifting focus>} //can be many
e.g., <4><2><P><AB>

Description
Only one sketch may be considered to as "having focus" at any one time. A sketch may be considered as "having focus" when the designers perform the following:

- when they're both looking at it
- when they point at it
- when they talk about it
- sudden head movements are a strong indicator
- if they start talking about it afterwards, a gaze is probably a focus shift

In this category, we are attempting to capture the moments in which the focus of the designers switch between sketches, as well as the reason that they shifted focus. In each of the videos reviewed thus far, the sketch that has a designer's focus can be identified by where the designer is currently writing, gesturing their hand, or the general direction of their entire body. Creating a new sketch should also be considered as "switching focus".

We are recording *all* focus shifts, essentially creating a continuous timeline that tracks the attention of the designers at all times. This goal leads to the inevitable question of "if there are two people at the board, how do we know who has focus?" In our coding sessions of the videos, one of the two designers nearly always took the lead in the conversation while the other designer silently followed their lead, or took the lead themselves by verbally responding. It was almost never the case in which the designers silently worked independently. In cases like these, they sat down and looked away from the board (causing the focus to be <no sketch>).

We were able to discern the active speaker by the following tendencies:

- if a person is speaking, then the focus follows them.
- if a person shifts their glance to a new sketch, it's not a focus shift unless the other person looks as well (unless the person is speaking).
- the goal is to capture the active sketch, and these rules are attempting to reflect that.
- if it's ambiguous, don't mark it.

A few more hints:

- the clearest indicator of focus is where the pen is writing, or hovering. That holds precedence over all else.
- track the gaze of the designers. Choose the best location of the focus. If a designer glances at a sketch, but has 'active' focus, record that as a switch as well, even if it only lasts half a second.
- designers that look away from the whiteboard or talk to each other is recorded. Record this as switching focus to no sketch <no sketch>. The designers' focus is only considered to switch to <no sketch> after talking for more than 15 seconds, but if they look back to the original sketch, then their focus is considered to not have changed. The timestamp of the focus switch is the moment that they look away. Any time spent addressing the experimenter, or replacing a pen out of ink, is immediately categorized as <no sketch>.
- if the coder cannot discern what sketch the designers are looking at, but the designers are in fact looking at the board, then code this as sketch <board>.
- if the designers are looking at the prompt, the target sketch of that shift will be coded as <prompt>. Do not record the shift type when moving to or from the requirements document. They must actually be reading the prompt, not just using it as a prop in an explanation.

Relationships between sketches:

Momentary reference (instantaneous)

Coding format

<momentary reference><active sketch><sketch being referenced>, <transition type> (can be many)
(5:06:0): <M-type><2><4><P><ALT>

If they do not return to the active sketch immediately, record it as <JT-type> (JT is short for juxtaposition, which was later changed to momentary reference in the journal article). If they return back to the active sketch, record that as: <JTB-type> (JTB is short for *juxtaposition-transition-back*).

```
<a>                 //a has focus
<JT-1><a><b><P>     //they move from a to b
<JTB-1><a><c><P>    //they move from b to c, and then return to a
```

If they move their hand over a set of sketches, we record that as a set of JTs. We only count the JT once in a sequence of gestures, we do not return back to it. Returning back to the active sketch resets the sequence, allowing for another set of JTs.

No momentary references may include <board>

Types of momentary reference:
<1> quick glance
<2> point with finger
<3> split focus by person with pen
<4> one designer glances somewhere else while their partner continues to maintain focus on the active sketch (can be either person with pen or person without). This type was not included in the final paper but should be included in the raw coding.
<5> split focus by person without pen. This type was not included in the final paper but should be included in the raw coding.

Hints

- repeated momentary references between the same sketches in a short period of time are indeed recorded as discrete, consecutive events
- momentary references are focus shifts that last less than 3 seconds (when it's single focus). You have to look at it for longer than three seconds, just pointing for longer than 3 seconds doesn't count.
- we're trying to capture what the active sketch is
- do not record instances of <4> and <5> which involve the prompt. The second person is outside of the video for over half of the total video time, meaning that we cannot record the majority of these transitions.

Description

In momentary references, a particular sketch will have the designer's focus, but they will reference other sketches. For example, a person may be writing with the pen, but momentarily look elsewhere. If they look for less than two seconds, we consider this a momentary reference.

After the reference, their attention should snap back. If, however, they gesture with their hand, even if only for a split second, then we consider this a focus shift. There is going to be many instances in which the designer will look around and it will be unclear where they are looking. In these cases, do not record it as a momentary reference. Overall, this is a difficult category to record, so simply try your best.
Purpose: To reference or compare sketches.

Level of abstraction

Coding format
(5:00:0): <4><2><AB>

Description
Abstraction refers to the level of detail that the sketch conveys, aka, this is a design definition of abstraction more so than a programming definition of one (making the definition more flexible).

A shift in the level of abstraction happens when the designer moves between sketches that represent a different level of abstraction of the sketches, and there is a clear hierarchy of information. The shifts that the designers make can be described as “diving into” a sketch, such as when the designers move from a sketch that represents a software component with many subcomponents, to a destination sketch that represents the detail of that sub component. Some examples of movement between levels of abstraction are:

- Moving between a sketch of a map and a sketch of an intersection
- Moving between a software architecture component with many components, to a sketch that focuses on just one of those components
- Moving between a software architecture component (aka a sketch of a box with title and elements) to a list that describes rules about that specific entity
- Moving between an architecture sketch or user interface sketch to a requirements list. The idea here is that the requirements list describes the design more abstractly than sketches composed of boxes-and-arrows or user interfaces.
- Moving between a class sketch with just class names and maybe some method names, to another class sketch with the fields filled out is considered a level of abstraction jump
- Moving between a sketch with many elements, to a sketch that focuses on just one element of the previous sketch is a shift in focus of abstraction

Alternative

Coding format
(5:00:0): <4><2><ALT>

Description
Shifts in alternatives represent movement between potential solutions. This is most evident by the speech of the individuals, i.e. they will verbally compare their old approach. Also, a redraw of

a previous sketch, even if the new sketch is only meant to beautiful the previous sketch, is also considered a movement between alternatives.
Purpose: Compare possible solutions and weigh their trade-offs.

Perspective

Coding format
(5:00:0): <4><2><P><R>

<P> - perspective switch between sketch types
<R> - perspective switch between sketch domains

Description
These are shifts within the same level of abstraction, and present another way to view the same information. In other words, the designer is moving around the different parts of the design. If the designer has a large sketch present with many parts, such as a user interface, their movements between parts of the UI are considered a perspective switch.
Purpose: Understand how the parts of a design fit into the whole, move between the parts of a design, or present a new view of the design that makes a particular view more salient.
Note: In the analysis we will be looking at the different kinds of sketches.

Point in time

Coding format
(5:00:0): <4><2><PT>

Description
Designers will sometimes create the same sketch, but each sketch will represent a different set of conditions or point in time. For example, one designer drew the same traffic intersection twice, with the first sketch representing time at *t=0*, and the second sketch representing time at *t=1*. In another example, designers drew the map as it appears while the user was editing it, and created another sketch to represent it while the simulator was running.

Purpose: In general, movements between points in time are used to help the designer in mental simulations, although this is not always the case.

Category 3 Codings - Activities they perform

Tags in this category may span over a period of time (continuous) or happen once (instantaneous). If a switch in sketch focus happens as an activity happens, attempt to code the activity “around” the switch. For example:

continuous
(4:59:0): <MS-S-1>

(5:00:0): <14><12><P>
(5:02:0): <12><14><P>
(5:05:0): <14><12><P>
(5:06:0): <MS-E>

instantaneous
(5:06:0): <2><4>, <P><ALT>

how they use sketches - This can belong to one of three categories:
<1> they do not use any sketches, they instead talk freely to themselves
<2> they gesture over existing sketches without drawing
<3> they draw or edit an existing sketch
<4> they create a new sketch to discuss the alternative

Mental simulation (continuous) - plays out over time

Coding format
(4:59:0): *<MS-S-1>*
(5:06:0): *<MS-E>*

Description
Mental simulation plays out over time, most likely by declaring the run time behavior, and simulates the user interacting with the design, or the data moving through the system. Mental simulations require a sequence of at least 3 events in the designer's explanation. Be careful not to include moments when they are designing the system. Mental simulations can be identified when the designers begin explaining how a design plays out over a period of time. Statements that are mental simulation have a clear cause and effect sequence that happens over a period of time (no matter how short).

Example of something that is not a mental simulation, but could possibly be confused as one: "No I think it would be there so that way you won't have to look over there once you've done it". In this statement, there is no passage of time. Another example, "It would be nice to allow them to copy settings from one panel to the next." This statement is a suggestion, there is no actual walkthrough
Purpose: Gain insight into the consequences of their design, understand how information flows among components, roleplay end-user input.

Examples
* if they deviate for more than 15 seconds, then MS stops
"Lets suppose..."
* An event is defined as a single user action, or the state of the program is changed (data passing from one object to another qualifies as changing the state).

Review progress (continuous)

Coding format
(4:59:0): <RP-S-1>
(5:06:0): <RP-E>

Hints

- *A stop phrase at the end of a review may signal that they're starting something new, and designing*
- verbally summarize their design (going down the list, or creating new items in a requirement list)
- plan what they are going to do next ("should we look at the sensor next? Sure, okay")
- discuss the requirements (but not actually say how they will implement it)
- mention that they have not done something yet
- whenever they discuss what their design needs to do (but they don't actually say how they're going to do it)
- * We do count instances in which they create the software list
- * Is stopped when they ask a question about a design item (not a clarification question), or discuss a design alternative
- * Another stop word: "We could...[proposes new design idea]"

Description
Simply put, these are all instances of speech in which the designers speak at a meta level about their design, including when they verbally summarize their design, plan what they are going to do next, and discuss the requirements. This includes all instances of looking at the requirements document, and also includes any discussion that flows out from the requirements document that includes sketches on the board. *Review progress* ends when the designers begin creating new parts of their design, or they ask a question about the design. For example, the designer will step back, or return to a list, and begin to list off items that need to be taken care of, often mumbling to themselves. In each case, the designer is taking stock of their current design, and checking that they have covered what they need with their design. If they begin designing new content, then this activity has ended.

If they are in the middle of reviewing the design, branching off to another sketch to understand the current design, and returning to a requirements lists afterwards is still considered as *Reviewing Progress.*

Purpose: Take stock of the design; understand what has been completed so far, determined what remains to be done.

//do we count instances when they restate assumptions about their design? My sense is that it depends on the case. If they are restating the assumptions directly aftering working on the part they are reviewing, then no, that is more of a design-level activity than managerial activity (which is what review progress belongs to), but if they are returning to many parts, then yes it is.

Discussion of alternative (instantaneous)

Coding format
(4:59:0): <DA-how they use sketch>
//"Write the first few words of the design alternative that they are proposing"
e.g. (4:59.0): <DA-1>

Hints
* A design alternative is a solution that competes or overrides an existing solution
* critiquing is a clue that a new DA is happening
* do not count it if the alternative was proposed previously
* there needs to be a concrete alternative that the new alternative can be compared to
* a refinement of a solution is not a new alternative

Description
This one isn't a design behavior, but it's important in supporting the category of shifting focus between alternatives. Our goal here is to capture moments when they are speaking about alternatives so that in the analysis we can say, "there were 30 instances of alternatives spoken, and of those, 20 involved the use of sketchs, and the rest were simply mentioned aloud."

* our threshold for marking a match in interrater reliability in this category is within 1 minute of each other's recorded event.

Some trigger words (not guaranteed DA, but very likely):

- "what I was thinking was..."
- "or it could be that..."
- "I thought that..."
- "Well, it seems to me that we can think of..."
- "Which one are we doing?"
- "or, we can say that..."