Investigating Domain Specific Visual Languages for Interactive Exhibitions

Andrew Stratton, Andy Dearden, Chris Bates
Cultural, Communication & Computing Research Institute
Sheffield Hallam University
A.Stratton@shu.ac.uk, A.M.Dearden@shu.ac.uk, C.D.Bates@shu.ac.uk

Abstract
Quando is a visual programming tool designed to enable Cultural Heritage Practitioners (CHPs), who do not have experience of computer programming, to create program scripts for interactive exhibits for galleries and museums. This paper reports preliminary findings from a qualitative investigation of the responses of CHPs to the Quando environment.

1. Introduction
In Cultural Heritage there is increasing desire to create digital interactive exhibits, intended to increase visitors' engagement and understanding and increase the ways in which public spaces are viewed and used. Adoption is seen as slow and limited with *'Over 60 per cent of arts and cultural organisations … constrained … by a lack of staff time and funding'* and *'over 40 per cent report a lack of key technical skills…'*, (Nesta, 2013).

CHPs typically commission experts to create software solutions; reducing the opportunity for CHPs to change/update the interactive experience and limiting their involvement.

Tools for CHPs
Current tools for creating digital interactivity are often technically demanding, especially for new technologies that might inspire visitors. Platforms, such as meSch, see Wolf (2015), are being created to support CHPs in specifying Interactive Digital Content without requiring programming language skills. Using meSch tools, high level 'recipes' allow CHPs to add/remove content and conditions to interaction description/s. This allows CHPs to create and/or modify the content of technically complex interactions at a cognitively suitable level, e.g. manipulating visual icons representing multi-media content, associating them with visitor data such as demographics or interests, e.g. the AtlantikWall exhibition, Museon (2016), requirements allow visitors to choose specific interests at the start of the exhibition and subsequently interacting with exhibits to trigger associated multi-media presentations. The meSch approach often requires a networked server for content delivery. Joining the recipes to the execution environment is dependent on high level technical skills, and the range of interactive behaviours available for a CHP is limited to the pre-defined library.

An alternative, lightweight, approach is proposed, where Quando, a set of visual 'block' based tools, allows CHPs to design interaction using a Visual (Domain Specific) Language in a similar approach to End User Programming, see Ko et al. (2011) for a summary. CHPs work at a higher conceptual level, concentrating on matching visitor interaction to content, i.e. using a domain specific language, see Fowler (2010). Quando seeks to offer a simpler approach for developers of the visual blocks that the CHPs will use. To explore this approach, CHPs were interviewed and asked to perform tasks in order to investigate the appropriateness of Quando in describing interactions.

2. Visual Language Design
The Quando language design follows an event matching approach. Rule blocks are created to describe events occurring in the environment of the exhibit. These events may represent temporal events (e.g. a regular event occurring every *n* seconds), or visitor triggered events (e.g. approaching a sensor). In order to respond to events, Rule blocks act as 'containers' for Action blocks; where the Actions are usually visible/audible effects for visitors, i.e. presenting multimedia content.

Quando uses Google's Blockly language (Frazer, 2016), which was designed for education purposes. Blockly has been chosen because, unlike Scratch, (Resnick, 2009), it allows customisation of blocks and allows output to different programming languages to be generated.

Quando blocks are intended to allow CHPs to describe behaviour visually, without learning a traditional programming language. The blocks are used to generate a textual programming script for execution on an appropriate platform. A library of Blocks is designed using terms that are intended to be familiar to CHP's rather than traditional imperative programming statements.

Using the current library of Blocks in Quando, a variety of visitor interaction behaviours can be described. These behaviours can be categorised by increasing interaction depth/complexity:

- **Static –** Fixed content is displayed
- **Sequence** - A regular, timed, sequence of content
- **Responsive** - Responding to an anonymous Visitor's choices, using a stateless model
- **Session** - Based on a Visitor's preference/demographic/interests/choices **during** a visit

Two further behaviours, not yet implemented, would support **Historic** behaviours using details of a Visitor's previous visit/s and **Inter-site** behaviour based on a 'partner' site's data for a visitor.

Prior to developing support for these more complex interactive behaviours, an investigation (see Section 3) has been conducted to explore whether a block based language would be acceptable for this user population, and to explore issues that might influence future development.

Quando blocks have focused on textual Identifiers, derived from the AtlantikWall exhibition interaction requirements, Museon (2016). The AtlantikWall requirements included video/audio/text and image presentations, for which Action blocks were defined, e.g.

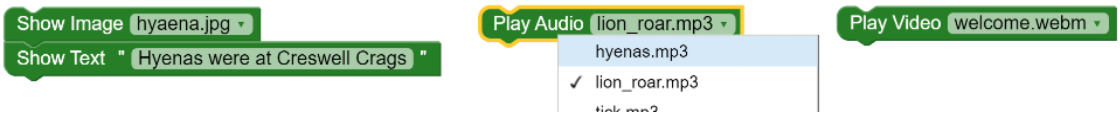

Figure 1 - Example Blocks with content selection by drop down list

Rule blocks were designed to distinguish between different types of event matching. Temporal blocks include 'Every' and 'Do for' blocks with CHP selected timings. Interaction Blocks were designed that show integration with recent technology, in this case through use of the Leap Motion controller (Bachmann et al. 2014) which uses two infrared cameras to detect visitors' inputs using touch free hand movement.

Note that the behaviour associated with an exhibit may be described by multiple independent rule blocks, each triggering actions when its conditions are met.

Figure 2 – Example Temporal and Interaction Rule Blocks

The Blocks were developed iteratively, starting with a device specific example script and resulting in a Block definition including interface configuration and a generative script, e.g.

```
var HAND_COUNT = 'hand_count';
quando_editor.defineRule({
    name: 'When Hands', next:false, previous:false,
    interface: [ {name: HAND_COUNT, title:' = ', number:1}, {statement:STATEMENT} ],
    javascript: function(block) {
        var statement = quando_editor.getStatement(block, STATEMENT);
        return "quando.hands(" + quando_editor.getNumber(block, HAND_COUNT) + ",\n"
            + " function() {\n" + statement + "});";
    }
});
```

Figure 3 – Example Javascript Block definition

3. Investigation

CHP Participants were recruited by recommendation and cold calling. The investigation focused on Participant's background, interactive use of the Quando tool and subsequent feedback. For interactive

use, twelve behaviours were used; seven Static/Sequence, two Responsive using the Leap Motion and three Session behaviours. For each question, participants were asked to interpret/edit the behaviours and rate their confidence in their answer. A semi-structured interview followed with participant answers recorded using audio and video which was transcribed after the interview.

An outline description of example behaviours is given below:

- Participants were shown the behaviour below and asked to recreate the same sequence in the editor. A paper printed copy was also visible to the participants.

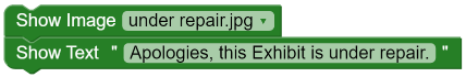

Figure 4 – Experiment 1.06 Sequence Creation

- The responsive behaviour below was shown and participants asked to modify the behaviour: *'Change the sequence to show the carving for 2 hands and the jewellery for 1 hand.'*

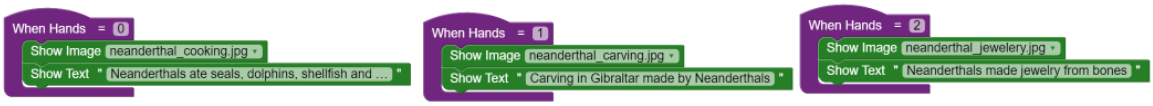

Figure 5 – Experiment 2.01 – Modify Responsive

Screen and Video/Audio was recorded of participants using the Quando editor in a full screen browser, to reduce navigation issues. The Leap motion was mounted at the top of the screen; reducing errors by increasing the distance from the participant.

4. Initial Observations

At the time of writing, five interviews have been conducted with CHPs using Quando for about twenty minutes. Two were experienced Curators, one a Museum Events/Exhibition Coordinator, one a Museum Software Developer and a Museum Researcher/Freelance Exhibit Designer. All CHPs have Postgraduate qualifications; three in Cultural Heritage, one each in Design and Software Engineering; only one of the participants had software development skills. Reviewing this early interview data, provides some interesting observations:

- Feedback was mostly positive - *'I'm really impressed with it. I want it'* and *'It would be brilliant for us to have this'*.
- Simplicity was commonly identified - *'once you've done it a couple of times, it's very easy to use'*, *'it's quite straightforward…once you understand the principles'*, *'(it's) quite logical'*.
- Some feedback was negative, *'it's quite alien to me…I'll immediately find a barrier when it's something…technological'*. The same participant was also positive about the touch free interaction - *'I liked the interactive element, the way it reflected ambidextrous behaviour was really good in getting across the point. I liked that…as for the others. I'm not sure'*.
- The colours were negatively identified with, *'I'm not too keen on the colours'* and *'don't like the colours'*. Blockly limits text within blocks to white text and has a restrictive colour model, saturation and value are shared across all and pastel colours cannot be used.
- Blockly platform behaviour caused difficulties with dragging and dropping blocks. A small yellow line is used to indicate the target of a drag; this needed explaining to most participants. Also, when blocks are attached in sequence, dragging a block will also drag any blocks 'attached' below it; most participants tried to undo this behaviour when it happened.
- When asked about programming styles for specifying common actions, participants universally disliked repetitive content descriptions, e.g. where the same action blocks are repeated within multiple rules – e.g. see below (where the Play Video block is repeated), though there was awareness of it's value - *'it depends where you're going with it next'*.

Figure 6 – Repetitive content across Rules

- The Rule/Action Blocks were mostly liked - *'I also like how the actions fit within the rules'*.
- Participants identified the benefits of the approach for non-programmers – *'for me with no IT training, it's really useful if I could setup digital interactive exhibitions'*, *'I feel quite confident that I could put something rudimentary together now…I haven't had to outsource and it means I can go and do it again'* and *'it's nice that you don't have to code it'*.
- There were concerns about how well the approach and the tools would scale to more 'complex' interactions – *'I wonder how difficult, how complex, you can get with it though'*.
- Children were identified as a likely audience, in particular the touch free interaction – *'that kids could interact with'* and *'it (leap motion) would work particularly good with schools'*.
- There was an expectation that training would be required – *'I would be quite happy to go on a training course'* and *'if they (museum staff) have the training'*.

5. Future Work

The use of visual tools for CHPs to describe visitor interactions is promising and worth pursuing. More complex behaviours need to be considered as well as alternative visual tools.

Further artefacts will be designed to explore CHP conceptual models, and appropriate authoring tools, of interest based interactions, including session, historic and geographic based interactions, i.e. interactions that depend on current and/or previous visits.

An analysis of the Quando concepts using Cognitive Dimensions (Blackwell et al. 2001) is also planned.

References

Bachmann, D., Weichert, F., & Rinkenauer, G. (2014). Evaluation of the leap motion controller as a new contact-free pointing device. *Sensors*, 15(1), 214-233.

Blackwell, A. F., Britton, C., Cox, A., Green, T. R., Gurr, C., Kadoda, G., ... & Roast, C. (2001). Cognitive dimensions of notations: Design tools for cognitive technology. *Cognitive Technology: Instruments of Mind* (pp. 325-341). Springer Berlin Heidelberg.

Fowler, M. (2010). Domain-specific languages. *Pearson Education.*

Fraser, N. (2016). Google blockly - a visual programming editor. *URL: http://code.google.com/p/blockly. Accessed May 16.*

Ko, A. J., Abraham, R., Beckwith, L., Blackwell, A., Burnett, M., Erwig, M., ... & Rosson, M. B. (2011). The state of the art in end-user software engineering. *ACM Computing Surveys (CSUR)*, 43(3), 21.

Museon (2016). The Hague and the Atlantic Wall. *URL: http://www.museon.nl/en/exhibitions/hague-and-atlantic-wall, Accessed May 20.*

Nesta (2013). Digital Culture: How arts and cultural organisations in England use technology. *URL: http://artsdigitalrnd.org.uk/wp-content/uploads/2013/11/DigitalCulture_FullReport.pdf*

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. (2009). Scratch: programming for all. *Communications of the ACM*, *52*(11), 60-67.

Wolf, K., Abdelhady, E., Abdelrahman, Y., Kubitza, T. and Schmidt, A. (2015), July. meSch: tools for interactive exhibitions. *Proceedings of the Conference on Electronic Visualisation and the Arts* (pp. 261-269). British Computer Society.