Overcoming Mental Blocks: A Blocks-Based Approach to Experience Sampling Studies

Daniel Rough
School of Computer Science, University of St Andrews
djr53@st-andrews.ac.uk

Aaron Quigley
School of Computer Science, University of St Andrews
aquigley@st-andrews.ac.uk

***Abstract*—Experience Sampling Method (ESM) studies repeatedly survey participants on their behaviours and experiences as they go about their everyday lives. Smartphones afford an ideal platform for ESM study applications as devices seldom leave their users, and can automatically sense surrounding context to augment subjective survey responses. ESM studies are employed in fields such as psychology and social science where researchers are not necessarily programmers and require tools for application creation. Previous tools using web forms, text files, or flowchart paradigms are either insufficient to model the potential complexity of study protocols, or fail to provide a low threshold to entry. We demonstrate that blocks programming simultaneously lowers the barriers to creating simple study protocols, while enabling the creation of increasingly sophisticated protocols. We discuss the design of Jeeves, our blocks-based environment for ESM studies, and explain advantages that blocks afford in ESM study design.**

***Index Terms*—Visual programming, end-user development (EUD), experience sampling method (ESM)**

I. The Problem - Programming

The Experience Sampling Method (ESM) is a research method over three decades old. It involves the repeated assessment of a study participant's behaviours, attitudes and states in their natural setting [1]. Traditionally ESM studies were conducted with pen-and-paper diaries, and an electronic signalling device such as a pager that would notify participants when to complete a diary survey. Today, ubiquitous mobile technology has motivated the development of smartphone-based ESM applications for alleviating researchers and participants from burdens associated with paper-based methods [2]. Such devices are an ideal platform to deliver personalised information to their users at opportune moments, or to collect reports on users' thoughts, feelings or physical symptoms without bringing them to a clinic or research laboratory.

In spite of their many benefits, smartphone ESM apps are still notably underused in relevant literature. We recently reviewed 148 publications published in the last three years with 'experience sampling method' or 'ecological momentary assessment' in their title. Of these 148 publications, while 56 used smartphone applications, 46 used PDAs, seven defaulted to pen and paper methods, and others had researchers manually call or text participants. This gives rise to our motivating question here, "How can researchers be empowered to exploit context-awareness, wireless connectivity, and multimodal input possibilities afforded by smartphones?"

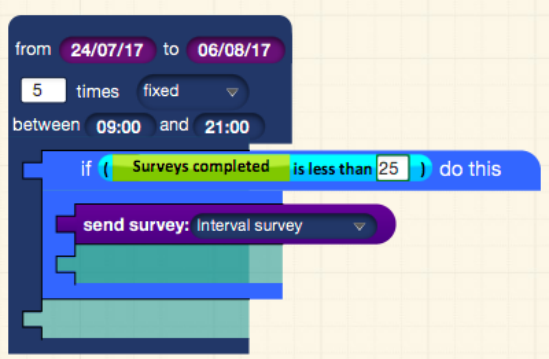

Fig. 1. Jeeves configuration for conditionally sending a survey

However, such researchers are not necessarily programmers, and do not have the time or ongoing need to learn, let alone develop custom applications for their studies [3]–[5]. This has resulted in efforts to develop free and proprietary tools for researchers to create and deploy ESM study apps. Our view is that for ESM the tools available are yet to strike a balance between high *usability* by non-programmers, and high *functionality* of the created apps.

II. The Solution - Blocks?

In existing work, we can see many examples of blocks programming being used as an effective approach for teaching computer science concepts to students of all ages within formal education settings [6], [7]. Through representing programmatic constructs as coloured jigsaw-puzzle pieces, this lowers the barriers imposed by textual programming languages such as Java or C++ that are used as introductory languages for novices. Blocks-based languages are also suitable for particular problem domains, where the domain-specific terms can be mapped directly into block representations.

Our thesis is that the qualities of block-based languages make them a suitable mode of communication for the domain of ESM studies. The feasibility of such an approach was validated in our previous work, where we studied the usability of our Jeeves environment by participants with a range of programming experience, including many with none [8]. In this prior study, 20 participants were asked to complete a series of tasks addressing the main functionality of Jeeves, over a 30 minute period. Data was collected through surveys, post-

study interviews and screen capturing of the interaction. A quantitative analysis of this data showed that the blocks-based approach achieved a high degree of usability among non-programmers. Broadly speaking [8], we found that participants were able to understand the concepts of blocks programming after a simple tutorial, were able to read and understand previously created ESM study configurations, and were even able to design their own configuration given a specific scenario.

III. Alternative approaches

We reviewed past and current ESM creation tools (hereby referred to as ESMCTs), and catalogued that many ESMCTs focused minimally on the usability of the tool itself. Despite their common goal to enable non-programmer researchers to create ESM study configurations, some still require researchers to edit text or XML configuration files [9], [10]. With no mediating user interface, the modification of large text files is cumbersome, prone to syntactical errors, and requires a researcher to learn appropriate configuration terminology.

Proprietary ESMCTs such as LifeData[1] and MetricWire[2] allow researchers to create configurations through web forms. Despite their simplicity, web forms have drawbacks that we feel render them inappropriate for the domain of ESM study creation. In particular, when complex scheduling protocols are employed, forms-based ESMCTs encounter usability issues in scaling up to represent these, particularly in the fluidity of making modifications.

Finally, ESMCTs (including the popular MovisensXS[3]) have used a flowchart-based visual programming metaphor, which has been successfully employed in educational visual programming languages such as Raptor [11]. However, due to the event-driven nature of experience sampling applications, we suggest that flowcharts are also a less suitable paradigm for modelling such applications, which we discuss in more detail in the next section.

IV. Proposed advantages

We have identified several advantages that we propose blocks programming has for ESMCTs. We draw comparisons between Jeeves, our blocks-based ESMCT, and the popular blocks environments 'Scratch' [12] and 'App Inventor' [13].

A. Blocks support event-driven programming

Blocks languages such as App Inventor support creation of event-driven applications, which respond to asynchronous external events. ESM study apps are also event-driven - participants are prompted at certain times of day to complete surveys, and may also self-initiate surveys by pressing onscreen buttons. In addition, smartphone sensing capabilities have now allowed events of interest to be automatically detected that initiate survey prompts.

Evidence from studies of Scratch and App Inventor in novice Computer Science education shows that students can create complex event-driven apps quickly, suggesting that blocks are an appropriate mental model for such a programming technique. 'Trigger' blocks in Jeeves are analogous in function, and similar in appearance, to the event-handler blocks of App Inventor. Likewise, our own evaluations of Jeeves suggest that participants understand the visual mapping of events to our Triggers.

Other ESMCTs use paradigms that may not support a researcher's mental model of their event-driven protocol. MovisensXS, arguably the current state-of-the-art in ESMCTs, uses a flowchart metaphor. Flowcharts are helpful for visualising programs that are executed from beginning to end, but do not map well to an event-driven model, where the execution is dependent on discrete external events. For example, a protocol might have surveys initiated at random times, or after a button press. These are separate events, occurring sporadically or not at all. Such an execution is ill-suited to a representation based on a linear program model.

Web forms are a familiar input mechanism for the majority of computer users, far removed from traditional programming. However, the guided, linear approach is problematic. In our user evaluations, participants would often define actions before deciding how to trigger them. Others would create triggers, then fill them with actions. A researcher's mental model may be inhibited by forcing premature commitment to either [14].

B. Blocks support sharing and reuse

Clear communication of ESM study protocols could be supported with a blocks-based representation. When ESM studies are documented in the literature, they are often ill-defined. In a workshop we conducted with Health Psychology students, we asked them to use Jeeves to model studies described in five publications. Their struggle with this task suggested that a common, formal language of communication is missing in ESM studies. It appears that natural language is insufficient for describing studies that could be rerun, a problem rife in scientific research [15]. In Jeeves, the blocks specification, survey questions, and user interface design, have an underlying JSON-like structure, which could be included as an appendix in publications to ensure adequate reporting.

At the blocks level, the readability of study configurations would also be useful for design inspiration. Learning by example and 'remixing' programs created by others are key elements of the success of Scratch [12]. Finding an existing project and tweaking it for one's own purposes could support efficient development, and collaboration between researchers. A blocks representation could also serve as a common artifact of understanding between researchers and participants.

C. Blocks support complex concepts

The research on education with blocks programming provides strong evidence that it is an effective method for delivering programming concepts. In Jeeves, we incorporate concepts from programming that could potentially incite confusion, such as conditional statements and variables that afford dynamic study configurations.

[1] https://www.lifedatacorp.com/
[2] http://www.metricwire.com/
[3] https://xs.movisens.com/

We have found conditional statement blocks are a useful means of representing 'combination' triggers - those that execute based on a co-occurrence of ongoing states and a discrete event. Huang and Cacmak have studied mental models of trigger-action programming, finding that participants have problems in disambiguating event and state triggers [16]. Making the difference clear at the user interface level, eliminates ambiguity. Consider Fig. 1. The specific time event is a trigger, and the number of surveys completed is an ongoing state condition. Multiple states could be supported by nesting these conditional blocks, or concatenating the conditional expressions. In an alternative approach, the ESMCT described in PartS supports complex triggers by joining events with arrows, but participants in a usability evaluation were not sure how events could be combined [17].

Our design of Jeeves requires that some notion of variables be implemented, to let researchers personalise apps to the individual using them. Fig. 1 shows a 'Surveys Completed' variable, which will be unique to each individual in an ESM study. In block-based environments, variables themselves are draggable entities, with properties such as shape, colour and text that serve as physical reminders of what they represent. This 'secondary notation' of Jeeves variables makes how they can be incorporated into the configuration visually explicit [14]. Thus, the variables are used like any other block, which we expect minimises the additional understanding effort through consistency. Neither the flowchart nor forms-based ESMCTs provide this functionality.

D. Blocks get the job done

Learnability of blocks programming and how well it supports users in transition to text-based languages is not an issue for researchers who just want to 'get the job done'. The authors of Scratch state that for users "who see programming as a medium for expression, not a path toward a career, Scratch is sufficient for their needs" [12]. With this in mind, researchers creating ESM studies simply need a means to express their desired study protocol as an application; they do not need to learn additional programming syntax and semantics. One of our qualitative study participants, a PhD student in Psychology, explained how he had been forced to spend time learning MATLAB, despite having no interest in programming.

In our studies of Jeeves, participants program as *bricoleurs*, observing the triggers, actions and conditions that they have at their disposal, and moving them around like real puzzle pieces [18]. While this allows more freedom of expression than forms-based environments, it has received criticism for going against traditional top-down software development. We do not aim to teach visual coding 'best practices' with Jeeves, we simply attempt to support intuitive practices.

Getting the job done also involves abstracting over complex functionality that simply needs to be used, not understood. Blocks encapsulate difficult concepts such as sensor data classification, making them easily accessible. For example, App Inventor has event handler blocks described in natural language, such as '*When Location changed*' or '*when Pedometer1.StartedMoving*'. Jeeves uses the same approach, defining sensed contexts at an abstract level to allow non-programmers to use this functionality in their ESM study. For example, a discussion with a medical Professor in our University elicited a need for prompting surveys at times *leading up to* as well as *following* a particular event. Thus, we encapsulated this functionality in a conditional expression block (Fig. 2).

Fig. 2. A Jeeves time expression (representing state rather than an event)

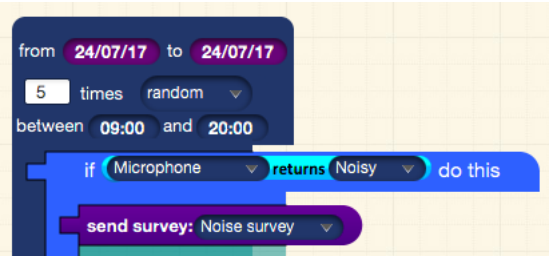

Fig. 3. A combination trigger, based on time and sensor data

Despite these advantages, we cannot claim that blocks are a panacea for all problems. For example, as users express desire for different functionality, how to incorporate additional blocks without overwhelming a researcher could be an ongoing issue. Also, not everything can, nor should it, be represented with blocks, and the level of abstraction required for different ESM concepts should be investigated.

V. Examples

To demonstrate the applicability of blocks-based environments to support real ESM studies, we provide two examples of functionality from previously described studies that cannot be replicated in other current ESMCTs.

A. Complex triggering

As previously explained, Jeeves supports 'combination' triggers by nesting conditional blocks (that check ongoing state) inside trigger blocks (that detect discrete events). In a study by Lathia et al. [19], these authors designed an ESM study motivated by previous use of audio data for investigating user behaviour. They developed an app that sampled microphone data at random intervals during the day, triggering surveys when noise was detected. Through the use of a trigger that fires at a specific time, and a condition monitoring background noise level, complex triggering behaviour is exhibited. This example is shown in Fig. 3, demonstrating simple block nesting that makes events and states visually explicit.

B. Variable usage

Representing an individual's information as variable blocks allows researchers to personalise studies based on that individual's particular attributes. We introduce this idea to participants in our ongoing qualitative study by demonstrating how a

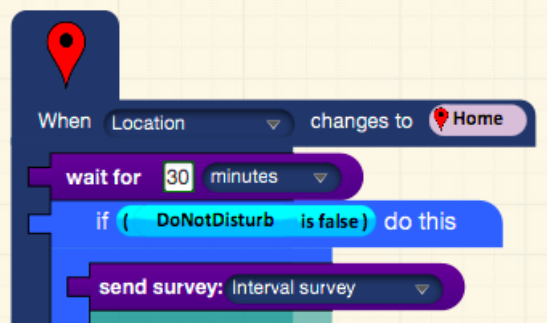

Fig. 4. Part of a Jeeves 'Do Not Disturb' specification

'UserIsHappy' variable can be used to conditionally prompt a supportive message if the value of this variable is false. To test their understanding, we ask them to model functionality hard-coded into many ESM applications - a 'Do not Disturb' button that mutes survey notifications. Participants have been able to model this behaviour, using conditional expression blocks and variable blocks, an example of which is shown in Fig. 4. This is an encouraging result that we hope reflects understanding of this complex functionality.

Although it is of great importance to easily enable traditional and simple ESM studies to be configured, providing researchers with the power to define complex functionality could allow them to fully exploit smartphone technology.

VI. Discussion

There are a number of implicit and explicit findings we suggest are important for the blocks community to consider. Events as first-order concepts in ESM are key for framing the use of a blocks language. Sharing and reuse, are core to support the thinking required by practitioners in developing ESM. In practice, sampling strategies can exhibit complex structures, which can be articulated with blocks. We suggest that the expressiveness of the visual block representation, abstracts to the level domain experts reason about their problems.

Our proposed advantages of representing ESM studies with blocks are based on evidence from studies of App Inventor and Scratch with novice students. In particular we note their ability to facilitate understanding of programmatic concepts, their affordance of sharing and reuse, and their support for creating highly functional apps with minimal effort. While we have demonstrated the possibilities that blocks programming affords ESM app development, we have not evaluated how, or even *if* researchers would utilise these possibilities.

Our research so far has validated the feasibility of using blocks to configure ESM study applications. Preliminary results of our qualitative study suggest variable blocks that represent individuals' attributes is a comprehensible concept by non-programmers and affords the possibility for researchers to configure study specifications for individuals that adapt to changes in their mental or physical state.

In future work, we will evaluate in both longitudinal and short terms studies how researchers use our blocks to create real ESM studies, and assess their attitudes towards our proposed advantages. We also hope to evaluate the usability differences in blocks, flowcharts and forms for constructing identical configurations, to address our hypothesis that blocks programming is the most effective, efficient and satisfying method for writing and understanding ESM apps.

References

[1] M. Csikszentmihalyi and R. Larson, "Validity and reliability of the Experience-Sampling Method." *The Journal of nervous and mental disease*, vol. 175, no. 9, pp. 526–36, sep 1987.

[2] V. Pejovic, N. Lathia, C. Mascolo, and M. Musolesi, *Mobile-Based Experience Sampling for Behaviour Research*. Cham: Springer International Publishing, 2016, pp. 141–161.

[3] M. J. McGrath and T. J. Dishongh, "A common personal health research platform-shmmer and biomobius." *Intel technology journal*, vol. 13, no. 3, 2009.

[4] Z. Skrba, B. O'Mullane, B. R. Greene, C. N. Scanaill, C. W. Fan, A. Quigley, and P. Nixon, "Objective real-time assessment of walking and turning in elderly adults," in *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*. IEEE, 2009, pp. 807–810.

[5] W. Hofmann and P. V. Patel, "SurveySignal: A Convenient Solution for Experience Sampling Research Using Participants' Own Smartphones," *Social Science Computer Review*, vol. 33, no. 2, pp. 235–253, 2015.

[6] M. Armoni, O. Meerbaum-Salant, and M. Ben-Ari, "From Scratch to Real Programming," *ACM Transactions on Computing Education*, vol. 14, no. 4, pp. 1–15, feb 2015.

[7] D. Weintrop and U. Wilensky, "To block or not to block, that is the question: Students' perceptions of blocks-based programming," in *Proceedings of the 14th International Conference on Interaction Design and Children*, ser. IDC '15. NY, USA: ACM, 2015, pp. 199–208.

[8] D. Rough and A. Quigley, "Jeeves - A Visual Programming Environment for Mobile Experience Sampling," *Visual Languages and Human-Centric Computing, 2015 IEEE Symposium on*, pp. 121–129, 2015.

[9] J. Froehlich, M. Y. Chen, S. Consolvo, B. Harrison, and J. A. Landay, "MyExperience: a system for in situ tracing and capturing of user feedback on mobile phones," *Proceedings of the 5th international conference on Mobile systems, applications and services*, vol. San Juan,, pp. 57–70, 2007.

[10] S. Thai and E. Page-Gould, "ExperienceSampler: An Open-Source Scaffold for Building Smartphone Apps for Experience Sampling." *Psychological Methods*, 2017.

[11] M. C. Carlisle, T. A. Wilson, J. W. Humphries, and S. M. Hadfield, "Raptor," *ACM SIGCSE Bulletin*, vol. 37, p. 176, 2005.

[12] M. Resnick, J. Maloney, A. Monroy-Hernández, N. Rusk, E. Eastmond, K. Brennan, A. Millner, E. Rosenbaum, J. a. Y. Silver, B. Silverman, and Y. Kafai, "Scratch: Programming for All." *Communications of the ACM*, vol. 52, no. 11, pp. 60–67, nov 2009.

[13] F. Turbak, D. Wolber, M. Sherman, F. Martin, and S. C. Pokress, "Events-First Programming in App Inventor," *Journal of Computing Sciences in Colleges*, vol. 29, no. 6, pp. 81–89, 2014.

[14] T. Green and M. Petre, "Usability Analysis of Visual Programming Environments: A Cognitive Dimensions' Framework," *Journal of Visual Languages & Computing*, vol. 7, no. 2, pp. 131–174, jun 1996.

[15] M. Munafò, B. Nosek, D. Bishop, and K. Button, "A manifesto for reproducible science," *Nature Human*, 2017.

[16] J. Huang and M. Cakmak, "Supporting mental model accuracy in trigger-action programming," *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing - UbiComp '15*, pp. 215–225, 2015.

[17] T. Ludwig, J. Dax, V. Pipek, and D. Randall, "Work or leisure? Designing a user-centered approach for researching activity in the wild," *Personal and Ubiquitous Computing*, vol. 20, no. 4, pp. 487–515, 2016.

[18] S. Turkle and S. Papert, "Epistemological Pluralism : and Voices Within the Computer," *Signs*, vol. 16, no. 1, pp. 128–157, 1990.

[19] N. Lathia, K. K. Rachuri, C. Mascolo, and P. J. Rentfrow, "Contextual dissonance: Design bias in sensor-based experience sampling methods," *Proceedings of the 2013 ACM international joint conference on Pervasive and ubiquitous computing*, pp. 183–192, 2013.