University of Groningen

Using TEMPEST

Batalas, Nikolaos; aan het Rot, Marije; Khan, Vassilis Javed; Markopoulos, Panos

Published in:
 Proceedings of the ACM on Human-Computer Interaction

DOI:
[10.1145/3179428](https://doi.org/10.1145/3179428)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
 Publisher's PDF, also known as Version of record

Publication date:
 2018

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Batalas, N., aan het Rot, M., Khan, V. J., & Markopoulos, P. (2018). Using TEMPEST: End-User Programming of Web-Based Ecological Momentary Assessment Protocols. In *Proceedings of the ACM on Human-Computer Interaction* (EICS ed., Vol. 2). (Proceedings of the ACM on Human-Computer Interaction). ACM Press. <https://doi.org/10.1145/3179428>

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Using TEMPEST: End-User Programming of Web-Based Ecological Momentary Assessment Protocols

NIKOLAOS BATALAS, Eindhoven University of Technology, The Netherlands

MARIJE AAN HET ROT, University of Groningen, The Netherlands

VASSILIS-JAVED KHAN, Eindhoven University of Technology, The Netherlands

PANOS MARKOPOULOS, Eindhoven University of Technology, The Netherlands

Researchers who perform Ecological Momentary Assessment (EMA) studies tend to rely on informatics experts to set up and administer their data collection protocols with digital media. Contrary to standard surveys and questionnaires that are supported by widely available tools, setting up an EMA protocol is a substantial programming task. Apart from constructing the survey items themselves, researchers also need to design, implement, and test the timing and the contingencies by which these items are presented to respondents. Furthermore, given the wide availability of smartphones, it is becoming increasingly important to execute EMA studies on user-owned devices, which presents a number of software engineering challenges pertaining to connectivity, platform independence, persistent storage, and back-end control. We discuss TEMPEST, a web-based platform that is designed to support non-programmers in specifying and executing EMA studies. We discuss the conceptual model it presents to end-users, through an example of use, and its evaluation by 18 researchers who have put it to real-life use in 13 distinct research studies.

CCS Concepts: • **Information systems** → **Web applications**; • **Applied computing** → **Psychology**;

Additional Key Words and Phrases: Ecological Momentary Assessment Method; Experience Sampling Method; Ambulatory Assessment; Software for Clinical Psychology; Organisation and Architecture;

ACM Reference Format:

Nikolaos Batalas, Marije aan het Rot, Vassilis-Javed Khan, and Panos Markopoulos. 2018. Using TEMPEST: End-User Programming of Web-Based Ecological Momentary Assessment Protocols. *Proc. ACM Hum.-Comput. Interact.* 2, EICS, Article 3 (June 2018), 24 pages. <https://doi.org/10.1145/3179428>

1 INTRODUCTION

Researchers working in diverse fields, which include the health sciences, the social sciences, human-computer interaction, and marketing and management, are increasingly in need of research methods that will allow them to collect data from study participants in situ and over sustained periods of time [58][52][10][14]. Such methods are growing in popularity thanks to the ubiquity of smartphones and Internet infrastructure [13]. With origins in different domains, they are known by different names, such as the Experience Sampling Method (ESM), Ecological Momentary Assessment (EMA), and Ambulatory Assessment. ESM is a method originally used in psychology, where participants are asked to regularly report their subjective experiences [26]. In EMA, a method originating

Authors' addresses: Nikolaos Batalas, Eindhoven University of Technology, Department of Industrial Design, Den Dolech 2, Eindhoven, 5600 MB, The Netherlands; Marije aan het Rot, University of Groningen, Department of Psychology, Grote Kruisstraat 2/1, Groningen, 9712 TS, The Netherlands; Vassilis-Javed Khan, Eindhoven University of Technology, Department of Industrial Design, Den Dolech 2, Eindhoven, 5600 MB, The Netherlands; Panos Markopoulos, Eindhoven University of Technology, Department of Industrial Design, Den Dolech 2, Eindhoven, 5600 MB, The Netherlands.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2018 Copyright held by the owner/author(s). Publication rights licensed to the Association for Computing Machinery. 2573-0142/2018/6-ART3 \$15.00
<https://doi.org/10.1145/3179428>

from behavioural medicine, the interest of researchers extends also to physiological processes [53]. The term Ambulatory Assessment is a collective term for such methods that additionally places particular importance on combining sensors and journaling to collect data about bio-psycho-social processes, involving people in natural settings [12]. As data collection typically spans an extended period of time, and because multiple measurements are taken, the aforementioned methods are also collectively known as Intensive Longitudinal Methods [22].

Compared to cross-sectional surveys that only provide static snapshots of processes in which respondents are involved, EMA methods afford a closer examination of processes as they unfold over time [38]. By supporting data collection in the natural course of life and in the context where events unfold, EMA methods can help researchers investigate contextual contingencies, reduce recall bias associated with retrospective self-reports, and provide a high degree of ecological validity. These advantages of EMA methods have sparked considerable interest in developing supportive technologies. Since their early inception as scientific tools in the 18th century [57], EMA studies have typically asked participants to record data with pen and paper, and often continue to do so in modern times [6]. However, researchers have always made efforts to incorporate their contemporary information and communication technologies into their arsenal: from using pagers [15] and alarm watches [28], to conducting the study solely on dedicated custom devices such as the Electronic Mood Device [29], or the Psymate [39].

The advent of affordable general-purpose handheld computers, first known as Pocket Computers, later as Personal Digital Assistants, and currently as smartphones, has encouraged researchers to develop specialised software to support EMA methods. Early EMA software was hardcoded for each specific study [54], but later systems allow generic protocols that can be customised to accommodate different studies. Advances in this area have focused on a number of features, such as the recording of multimodal data and background logs, the aggregation of data into a database server, and deployment on devices owned by respondents, so that they are not expected to carry a second device with them, and so that costs are kept low.

As software platforms gradually converge towards a common set of technologies used (e.g. client-server applications for the administration of studies) and common basic features for collecting data (e.g. in terms of interfaces served or capturing context), it is also useful to pay attention to how an EMA platform can be put together. This paper puts forward a set of abstractions for organising an EMA platform that is programmable by researchers. These abstractions serve to conceptualise EMA protocols as programs, and allow versatility in their composition. TEMPEST is an implementation of the proposed organisation. We describe how it addresses end-user programming concerns, and we detail its components. The adequacy of the organisation of TEMPEST for the end-user programming of EMA protocols is demonstrated through 13 deployment cases, where researchers used TEMPEST for a variety of EMA studies, and we discuss our experiences. It is emphasised that these case studies concerned actual research projects, where the researchers opted to use TEMPEST of their own accord. They were not field studies where the researchers would use TEMPEST primarily in order to evaluate it. Finally, we report responses provided by researchers to a summative evaluation involving a technology acceptance survey.

2 RELATED WORK

In contrast to the great amount of data collection efforts that consist of probing respondents only once (usually by distributing a questionnaire), EMA studies demand prolonged involvement of participants. As such, their management and execution can be a challenge even for seasoned researchers [11]. Software systems, besides helping collect and store data digitally, can also include automation that eases the burden of managing an EMA study, by taking the longitudinal nature of the effort into account and providing features to support it: e.g., recruitment, briefing, tracking

individual progress at different phases of the data collection, signalling participants to report data, changing the questionnaire items depending on the phase of the study, providing relevant instructions at the right time, etc. Systems that are built for the composition of forms for general purposes, such as SurveyMonkey [25], Google Forms or LimeSurvey [49] do not, to our knowledge, offer or emphasise such features. In theory, researchers can appropriate simple survey tools, in ways similar to using paper forms, in order to carry out an EMA process, e.g., by repeated administration of a questionnaire as each participant goes through the study and by "manually" aggregating data to create a temporal data record. In practice however, such an approach can be laborious and prone to errors, and is not appropriate for more dynamic data collection protocols that would depend on time or sensed context.

Apart from general-purpose survey tools, there have been numerous software applications intended for longitudinal data collection that cater only to a very specific type of study or protocol. Examples are applications such as Toss'n'Turn [37], dedicated to sleep quality evaluation, or such as iHabit [48] that target self-awareness. Literature often discusses how well particular pieces of software serve their specific domain as in [20], or [46]. However, the sampling protocols in such applications are hard-coded, and cannot be reconfigured or appropriated by researchers for different types of EMA studies or across different domains.

The Experience Sampling Program (ESP) [2] was amongst the first to realise general-purpose, digital, experience sampling studies, allowing researchers to configure custom questionnaires on a Personal Digital Assistant (PDA), and eliminating the need to program purpose-specific applications. However, installing the system and initialising a survey were still challenging activities that required proficiency in software and took up considerable time, even running up to weeks for a single study. The software would be installed as a native application on Palm Pilot or compatible devices; this required effort for configuring the system on the available devices, and necessitated the provision to respondents of the device that would run the software, rather than allowing them to use their own device. Importantly, the PDA's limitations for input forced ESP [2] to only offer scale-type items for users to report data with. Last but not least, data could be extracted from a device only after it had been returned to the researcher at the end of the study. This could be inconvenient from the researcher's perspective as participants might fail to adhere to the sampling protocol e.g., because they did not understand instructions, they neglected or were not able to provide self-reports, etc. In such cases, the researcher would only find out after the sampling period had ended, when it would be difficult or costly to start again.

ESP [2] was essentially a computerised conversion of paper-based methods. However, the proposition for Context-Aware Experience Sampling (CAES) [31] in 2003 was one of the first instances that considered the Experience Sampling Method as native to the digital realm. Specifically, CAES extended the automated signalling of participants to occur not only on the basis of time, but also based on processed data streams, provided by the sensors of participants' devices, reasoning about the context they find themselves in. MyExperience [23] in 2007 was probably the earliest system to implement this proposition, for phones running Windows Mobile. It allowed access to sensor data streams for context-based triggering of actions, configurable either through C# programming, or through XML scripting. It also included server components that the phone could synchronise with, to download XML scripts, or to upload stored data.

Currently, smartphones have been widely adopted in developed countries, and adoption is rising in others, where wired communications infrastructure is not extensive. Wireless Internet connectivity is widely available, so exchanges of data can take place in real-time. For programmers, the tasks of setting up digital forms or collecting data on a server, are more straightforward compared to the past [32][21]. With regard to data collection, the focus has largely shifted towards reducing replication of effort and packaging more functionality into software libraries that EMA system

developers can use to build their own applications; examples of efforts toward this direction are the Aware Framework [19], the Funf framework [1], ResearchKit [27], ResearchStack [17], Survalytics [41], and Purple [50]. In parallel, efforts have concentrated on making easier to manage, turn-key solutions available; Ohmage [44], Sensus [59], and numerous other commercial applications have appeared, which offer graphical interfaces for composing EMA studies. Paco [18] offers a code library that EMA application developers can extend, and also a composition interface where researchers can compose surveys as short single-page questionnaires. Jeeves [47] focuses on offering a scratch-style [34] visual programming interface for specifying conditions that trigger sampling sessions.

It can be difficult for any single piece of software to meet the exact needs of every researcher, especially as requirements and demands evolve with time. Marketplace dominance shifts from one operating system to another, newer hardware that needs to be supported is frequently introduced, and the needs of researchers evolve. It should be acknowledged that software platforms for EMA studies try to be configurable, so they can be useful in diverse contexts, and also try to be extensible, so they can acquire new features or respond to evolving requirements.

This paper does not attempt an exhaustive account/comparison of features that different platforms do or should offer. We recognise that priorities of their developers are very different as they address diverse technological and application contexts. For a discussion on specific features that certain EMA software applications provide support for, the reader can refer to [47]. It is indeed the case that EMA applications tend to converge to a shared set of technologies and features, that at the very least can replicate paper-based protocols with conditional presentation of certain questions, and can allow the remote administration of a study.

In this paper we are concerned with how an EMA software platform can be organised, so that protocols can be composed like platform-independent computer programs, by the researchers themselves as end-user developers. End-user development refers to the case where the system developers provide a programming environment that allows end-users to create their own software solutions, addressing needs specific to their problem that cannot be known to the developer [42]. In this case, EMA software developers provide the tools to allow researchers, who are not necessarily software developers, to perform EMA studies, by describing their procedure for data collection (i.e. protocol) in terms of abstractions that are relevant to the data collection effort itself, rather than the functionality of the underlying platform.

In the following section we discuss the system's main components and provide a basic example of its use that illustrates how these components make it programmable for EMA studies.

3 ORGANISATION OF TEMPEST

3.1 Requirements

Initial work on TEMPEST sought to satisfy sets of requirements as they had been proposed in literature. From the aspect of functional requirements, Khan [32] and Fischer [21] offer similar recommendations. In his recommendations for the design of ESM tools, Fischer [21] proposes the following:

- “Think client-server”: A server can act as a central repository for devices to draw configurations from, which the researcher will produce only once.
- “Design for authoring”: An authoring interface can make the creation of studies easy and efficient.
- “Make use of people’s own devices”: When researcher-owned hardware is assigned to participants, extra costs and effort are incurred, both for the researchers and the participants.

- “Design for different levels of study complexity”: Different implementation options for the study would allow a researcher to choose between e.g. a browser-based client, that is easy to distribute and requires no installation from participants, versus a native client for more sophisticated functionality.
- “Make wise client choices”: Be aware of the freedoms and restrictions that your choices in technological platforms have, e.g., interoperability vs platform-specificity.
- “Support orchestration”: Allow researchers to monitor the study, adjust its contents, offer help to participants or even expel them when necessary.

Khan [32] proposes similar features, by arguing for making use of participants’ phones, designing for quick and easy installations on mobile devices, and allowing data on a device to be automatically synchronised to a remote server.

In addition to satisfying the aforementioned requirements, TEMPEST aims to enforce a separation of concerns between different software components [3]. In this way, problem-owners can be given tools to cater to their own domain, which can evolve according to their needs, independently of other application components. The composition of a study protocol should be controlled by the researcher conducting the study, low-level mobile services should be the concern of a mobile developer, and server-side database features should be catered to by a database developer. This separation of concerns can ensure that these different platform components can be interchangeable, and in doing so promote the platform’s extensibility and prolong its relevance to the field.

Another initial goal for the system has been to extend sampling interfaces beyond typical form elements, and this has been implemented through *Widgets*, which are described in the next sections. Other particular components detailed in the following sections, that help cast EMA protocols as programs, came about after other initial attempts did not prove versatile enough in expressing protocol configurations. An early attempt [4] cast the EMA protocol as a hierarchical tree that had short questionnaires as its nodes. Branching to either sub-tree could take place by asserting the value of user-input specific to that particular node. The implementation of a memory store allowed us to perceive each input interface as corresponding to a variable declaration in a program, and in turn questionnaires could be thought of as corresponding to function calls in a program. Enabling their sequential and conditional execution resulted in the system under discussion.

3.2 Components

TEMPEST is a client-server system, friendly to smartphone deployments, built for in situ data collection (Figure 1). It is comprised of the following components:

- **A server** that stores the protocols produced by researchers and the data reported by participants.
- **A web application client for researchers**, with a WYSIWYG (What You See Is What You Get) interface, for preparing a data collection protocol, which can be deployed to a cohort for execution. The protocol is stored as a JSON object that specifies content and parameters for object supported by the system’s runtime environment, namely *Widgets* and *Screens* (Figures 6a and 7a), *Conditionals* (Figures 6b and 7b) and *Sequentials* (Figures 6c and 7c). Also, it allows management of the participants (Figures 8a and 8b).
- **A client for participants**, that is made available either a standalone web application, or a native application that embeds the web-based client. It downloads the JSON protocols produced by researchers. The client features a runtime environment that executes the protocol, by instantiating JavaScript objects and bringing into view HTML elements, as per the JSON configuration. The types of objects that the runtime environment supports are:

Fig. 1. EMA researchers use a web-based WYSIWYG environment to edit an EMA program, as a set of structures called *Assignable Items*. Participants use a web-based runtime environment with an optional native layer, that interprets the *Assignable Item* configurations and runs the EMA program.

- **Memory** : It is an object that stores named variables, that can be freely written and read, even through arbitrary scripting. It is persistent across sampling instances (sessions), and maintains a history of values for each variable. In this way it allows the execution of protocols which have logic that spans over time and sampling sessions, e.g. “ask a question in this session, if a particular answer had been provided in a previous session”.
- **Widgets** : They represent sampling instruments or pieces of functionality. Examples would be a text-field, a rating scale, an interactive cognitive test, a custom script, or a system function call that provokes an action or produces a data value. *Widgets* have names that correspond to the variable names, the values of which are stored in Memory.
- **Screens**: They are lists of *Widgets* that help organise them on a single page.
- **Conditionals and Sequentials** : They provide the execution logic for the protocol. *Sequentials* arrange items for execution one after the other, while *Conditionals* determine what items to execute next, based on the evaluation of a logical statement. Those items, can be either *Screens*, or other *Sequentials* or *Conditionals*. All three item types belong to a superclass that the system calls *Assignable Item*.
- **Built-in functions**: Built-in functions (e.g. date and time, random number generation, count of days since a study started) allow the articulation of the EMA protocol at a level of abstraction and in a terminology targeting the researcher/domain expert and making transparent technical issues relevant to their implementation.

3.3 The EMA protocol as a program

The runtime environment provides sequential and conditional execution of EMA protocol components (*Assignable Items*), and allows the storage and retrieval of named variables. Essentially, it performs the tasks that a basic computer would perform to execute algorithms. Consequently, the runtime environment allows us to think of EMA protocols, in their entirety, as programs that the user specifies through a graphical user interface in TEMPEST.

The conceptualisation of an EMA protocol as a program suits the stepwise, algorithmic nature of a longitudinal sampling process. Protocols already feature computational processes as part of the data collection (e.g. sampling a sensor). As has been argued also in the end-user programming of awareness systems [36], one can think of human participants as asynchronous processes that, prompted by an interface on screen, manipulate that interface and provide values as responses to the issued requests.

No other assumptions with regard to human behaviour are made, other than that an interaction with an interface component (e.g. a button press) will take place at an unknown time. Thus, from a program's standpoint, humans act similarly to a server computer, a GPS sensor, or any other asynchronous process. In any of these cases the program issues a request to the asynchronous process and a received response triggers further execution via callback functions. The time it takes the process to respond may be indefinite and in some cases a response might never arrive. The program can choose to execute a different process while waiting for the response and resume execution without a response after a timeout period, or can treat the participant as a blocking process, where execution does not continue until that process has ended.

By understanding a user's manual interactions in this way, we can model human-provided input in the same way as computational processes; both are initiated in the same way (a user interface is rendered or a function is called), and both terminate in the same way (signalling an end-event to the structure that initiated them). Consequently, manual and automated data collection components can be treated as interchangeable items, or as parts of a sequence of instructions that the researcher issues, i.e. a program.

In composing such programs, TEMPEST raises the level of abstraction for a researcher, so that they need not be concerned with what process answers their requests for data. *Widgets* are functional units that encapsulate (potentially complex) functionality, can be treated as self-contained components, and can be used in an EMA protocol in a uniform manner. The EMA program that the researcher then composes is concerned with parameters that have to do with the experiment itself, and not with low-level programming of the tools used for the experiment.

3.4 Implementation

As far as implementation is concerned, the flow of execution between components such as *Screens* in a sequence, is managed through the browser's event model. Nested structures notify their parent-containers of their change of state, mainly about the end of their own execution. Parents can then instantiate and delegate control to the next appropriate nested structure. Also, for the persistence of memory across sampling instances, the browser's `localStorage` object is used.

While a browser-based implementation is sufficient to achieve platform-independent execution for diverse data collection scenarios, there are cases where device-specific functionality is desired. In such cases, a JavaScript Bridge is used, where a JavaScript object acts as a proxy for native function calls, and inversely, native code can initiate execution on the web-browser component level (such as `WebView` on Android), by sending it strings that are then interpreted by its JavaScript engine. As browsers continue exposing new APIs to allow access to system components, such as GPS, accelerometers, or even Bluetooth, native layers of implementation tend to become less necessary.

4 EXAMPLE SCENARIO

We present a simplified EMA scenario derived from an actual use case of TEMPEST. We thus aim to illustrate how one might put the system to use, and in doing so, introduce the various system components as they come into relevance¹. In our scenario, a researcher builds a short questionnaire for an event-contingent experience sampling protocol, where the participants are expected to provide two pieces of data, upon waking from sleep:

- Indicate degree of agreement to the statement "I slept well" using a slider between 0(agree) and 10(disagree). We will call this variable `sleepSatisfaction`.

¹For clarity in Figures 2a, 2b, 4, we use faithful vector representations of interface artifacts instead of bitmap screen-captures. Figures 3a, 3b, 3c and 9 are in UML notation

Fig. 2. *Screens* can be composed of separate *Widgets*

- Indicate agreement to the statement "I feel energetic" by choosing yes or no as an answer. We will call this variable `feelsEnergetic`.

Also as part of the protocol, and for managing the study, the researcher has assigned an ID number to each participant, which they are expected to provide in order to uniquely associate their responses. Finally, a "Thank you" note signifies the end of the sampling session. The following sections outline how one might build this questionnaire in TEMPEST, and the concepts involved in the process.

4.1 Widgets and Screens

The first task is to arrange the interface units that the participant will be interacting with, and which produce the variable data records, onto a single page for presentation. These units are the *Widgets*, and the page they are arranged onto, is the *Screen*. Figure 3a indicates the relationship between *Widgets* and *Screen*.

4.1.1 Widget. The *Widget* is a unit of computation. It can be a user interface, but also a function that computes a value, or runs a script, which is not rendered on the user's screen. When it has a value to return, it produces a data record, which is a key-value pair of the *Widget*'s name and its value. Figure 2a presents two examples of *Widgets*.

4.1.2 Screen. The *Screen* is composed as a list of *Widgets*. It is one kind of an *Assignable Item*, which means that it can be assigned for distribution to a participant. There are also other kinds of *Assignable Items*, *Conditional* and *Sequential* items, which are discussed below.

The researcher in our scenario composes a *Screen* called `Scr_Sleep` out of two *Widgets*, to which she assigns their respective variable names, `sleepSatisfaction` and `feelsEnergetic` (Figure 2b). Upon submission, the record produced is committed to the system's memory store. Records in memory are addressable by their *Screen* and *Widget* name, as `<Screen Name>::<Widget Name>`

Fig. 3. UML representations of *Screens*, *Sequentials*, and *Conditionals* as derived from *Assignable Item*

Fig. 4. The three consecutive *Screens* the participant will be interacting with. On the first one, the text-input *Widget* produces a value for the variable *pID*, which can be addressed as *Scr_ParticipantID: :pID*. Each *Screen* can be viewed as a tree structure where the *Screen* is the root node and *Widgets* are leaves.

4.2 Sequences of Assignable Items

Potentially all data records in a protocol can be arranged onto a single screen. However, the subdivision of the data records to be collected into different screens, allows easier management of these assets. *Screens* can be strung together into distinct sequences, allowing also greater versatility in protocol execution.

4.2.1 Sequential. A *Sequential* is a type of *Assignable Item* (an item that can be assigned/distributed to a participant) that can contain an ordered set of other *Assignable Items* (Figure 3b). These could be *Screens*, *Conditionals*, or other *Sequentials*.

The researcher in our scenario composes the other two *Screens* in the protocol, the first *Screen*, requesting the participant's ID, and the last *Screen*, thanking the participant (Figure 4). All three *Screens* can be placed into a *Sequential Item* (Figure 5), which can eventually be assigned to participants.

4.3 Conditional Statements

On certain occasions, the execution of a protocol needs to vary according to a specific parameter. For example, the day of the week or the hour of the day, the participant's gender or age, or a specific response at a certain question could all be used to modify how the sampling will evolve for each participant.

4.3.1 Conditionals. *Conditionals* are types of *Assignable Items* that consist of a logical statement, an *Assignable Item* to execute if the statement is true, and an *Assignable Item* to execute if the statement is false (Figure 3c). Logical statements can make use of the data records as produced by

Fig. 5. Final protocol as a nested tree. Execution of the first *Screen* is conditionally performed by the *Conditional Item* which contains it. When the logical statement `Scr_ParticipantID:pID==null` evaluates to true, our *Conditional Item* executes the *Assignable Item* `Scr_ParticipantID`.

Widgets in Screens. Also, they can make use of variables that are declared as part of the state of the current participant (see section 3). Finally, they can reference internal system functions that return values, such as `SystemTime`.

In our scenario, a *Conditional Item* can be configured so that the *Screen* that calls for the participant ID only be shown once, at the first session of the study. Every other sampling instance should be still be associated with the same participant ID.

The *Conditional Item* `Cnd_CheckForPID` (see `Cnd_CheckForPID` box in Figure 5) checks the system’s memory for the value of the variable called `Scr_ParticipantID::pID`. If the memory reports null (there is no value) then `Scr_ParticipantID`, the *Screen* that requests the `pID`, is shown to the participant, otherwise execution continues as specified by the sequence.

As a last step, the researcher would need to update the original *Sequential Item*. Since the *Screen* `Scr_ParticipantID` is conditionally executed by the *Conditional Item*, the *Screen* needs to be removed and replaced by the *Conditional*. Figure 5 shows the updated *Sequential Item*.

4.4 Participants and Groups

Fig. 9. *Participant Groups* contain multiple participants and can reference multiple *Assignable Items* to make up the study’s protocol.

To manage participants, the researcher creates groups and assigns individual *Participant* objects to each *Participant Group* (Figure 9). Each *Participant* will have its own URL to the study and the user it represents can access the protocol via the Web if the researcher wishes them to, or use the corresponding user account to access the study via the native mobile application. The researcher assigns the study protocol to the *Group* by selecting one or more *Assignable Items* (Figure 8b).

The researcher also has the option to distribute a group’s protocol as a single URL to a cohort, for convenience or in the case that its members are not yet known. In turn, they can access the study anonymously and the system will automatically construct user accounts for those who participate, and populate their corresponding participant group accordingly.

(a) Previews of *Screens*, each with its *Widgets*, as they would appear on a mobile screen. In this view, *Screens* are presented as a collection of assets.

(b) *Conditional* statements are if-then-else structures that specify a logical proposition and the *Assignable Items*, to which execution should flow, when the proposition evaluates to either true or false.

(c) Two *Sequential* structures, configured as lists of other *Assignable Items*. *Screen*, *Conditionals* and *Sequentials* share configuration and properties of all *Assignable Items*

Fig. 6. The researcher can build three types of *Assignable Items*.

(a) Configuring a *Screen*. In the lower portion of the screenshot, a highlighted textfield *Widget* exposes its properties.

(b) Configuring a *Conditional*. The user can compose the proposition and choose *Assignable Items* for execution to flow to.

(c) Configuring a *Sequential Item*. An ordered list can be composed out of all available *Assignable Items*.

(d) Setting a *Time-Trigger*. In this particular portion of the interface, the dates on which the trigger will fire can be chosen.

Fig. 7. Editing protocol components

Participants can also have property-variables, e.g gender, age, start-date, etc. This allows the researcher to preset parameters without asking participants for input, and make the sampling protocol contingent to those, personalising it to each individual. Participant properties are addressable in the same way that *Widget* values are, for example if a participant is given the property age then it would be addressed within the protocol (e.g. in scripts or logic statements) as `Participant::age`.

Last but not least, participants can be either in enabled or disabled mode. In this way, the researcher can choose to prevent access for certain participants, or allow it again later, according to the needs of the data collection.

(a) Participant groups. Each member is a unique user/participant in the system. Dynamic groups can also be generated by the system. Here, allParticipants would contain every participant in every group.

(b) Assignable Items are assigned to groups rather than individual participants. Here, for group heezeEnglish, Q_ENGLISH has been ticked.

Fig. 8. Managing participants

4.5 Protocol execution

Researchers have the choice to build event-contingent protocols, where the sampling process is initiated by the participant themselves, based on an agreement with the researcher on what constitutes the event to cause the initiation. Such protocols can be executed by TEMPEST on its web-based client, which does not require participants to install any additional software. Instead they can visit a link to a web-application that can also function offline. All that is needed to get started is for researchers to distribute link to participants.

Researchers can also build signal-contingent protocols, in which the smartphone notifies the participant that a sampling session should take place. For these studies, participants are required to install a native application on their Android smartphones, which can create notifications on the device and execute the protocol. Notifications can potentially be triggered by any sensed event, e.g. entering a geographic location as sensed by the GPS, but time is the most common trigger, e.g. prompting the participant for input at midday every day. In the cases where the system was deployed, triggers more complex than time had not been needed, and further diversifying the types of triggers implemented in the system had not been an immediate priority.

Researchers can add *Trigger* attributes to any *Assignable Items*, that instruct the device how to trigger it, when it has been directly assigned to the participant group. TEMPEST currently supports time and location triggers. The execution takes place in a web browser embedded into the Android application.

4.6 Collected Data

Collected data can be compiled into a comma-separated values (CSV) or excel file, where each row represents a single sampling session, and each column one of the variables declared as *Widgets*. Aside from the whole data set, separate files can also be downloaded separately for each participant.

Researchers can import the data into their software package of choice (e.g. SPSS) for further processing.

5 VALIDATION OF TEMPEST THROUGH ACTUAL USE

This section discusses insights gained from a formative evaluation based on 13 real-world deployments of TEMPEST (Table 1), which primarily aimed to collect requirements and feedback from researchers. These were studies carried out by experts in different fields. The researchers created their own protocol using TEMPEST. In all cases the creator of TEMPEST was available for explanation and troubleshooting. All studies reached completion, and researchers were able to fulfil their goals for data collection. All studies except D6 and D12 made use of the participants' own devices. In cases where custom interfaces for data input were needed, the creator extended the library of *Widgets* to satisfy those requests.

Extensions to the library of *Widgets* can be made in the following manner: Every *Widget* is built on a basic common template that performs two functions. One function is the rendering function, which renders interfaces that capture data (or executes code that generates data, in the case of automated ones that do not demand user input). The other function is to submit those pieces of data to its parent structure for further handling (in these cases, a *Screen*). New *Widgets* can be produced by reusing this basic template and providing the implementation for the rendering function, in JavaScript.

In the summaries in Table 1, the size and duration of the studies refer to the cohorts and datasets that contributed to the results of the study, as reported by the researchers themselves. The size and duration extracted from the usage logs was greater, as it often involved activity auxiliary to the study itself. For example, numerous participant accounts and collected data are only meant for testing and piloting, or it had been the case that researchers had cleared data from the online database, keeping only their own local records, after the study and analysis was finished.

In each case, the researchers recruited participants and carried out the studies in their own time, at their own places of work, as they would normally do, without our supervision. In doing so, in certain cases or for certain parts of each study, they also had to delegate work to other collaborators, whom we also had no awareness of. Eventually we asked each of our own contact persons to fill out a questionnaire and ask of their collaborators to do the same.

In the sub-sections that follow we outline why TEMPEST was chosen for the studies mentioned, and how it evolved by being put to use. We also provide details on feedback that the researchers offered, and how the system was evaluated in one of the studies (D4) by participants themselves. We also offer a closing sub-section with a summary and our reflections.

5.1 Reasons for using TEMPEST

It is not possible to know how researchers might have appropriated other software, or might have shaped their approach to their studies in the absence of TEMPEST. We can however point to principal features that researchers found desirable in the software. As researchers reported (Table 3, they did not have extensive experience with other systems. Therefore the following should not be considered as features mentioned in comparison to other systems, but rather as desirable features that offered an advantage over alternatives that the particular researchers had already considered.

5.1.1 Event-Contingent Recording. Studies that involve Event-Contingent Recording (ECR) are often executed with pen and paper. In D4, D5 and D11, the researchers reported that the use of TEMPEST offered several benefits; the number of entries per day was not constrained by the physical size (number of pages provided) of the diary, they could monitor the data collection while it was going on, and they did not have to transfer data manually from paper to digital spreadsheet.

Deployment	Field	Type	Summary
D1. Integration with app-store deployed idAnimate [43] to sample intentions for using software features [5].	interaction design	(event)	Over the course of 602 days, a population of 4157 unique users provided a total of 8316 sessions. From the total 4157 who had been sampled at least once, 2425 were sampled a second time while using the application, 187 users for a tenth time, and 10 users for a 50th.
D2. Combination of TEMPEST with AIRS [55] for context-based triggers of surveys [9].	computer science	(signal)	The resulting system was tested with 18 participants in 3 different situations. 42 questionnaire items were triggered based on context.
D3. Integration with app for location based sampling [30].	media research	(signal)	23 participants provided 107 responses over the course of 4 weeks, evaluating ads triggered by location.
D4. Social Interactions across the menstrual cycle in women with self-reported PMS [7].	clinical psychology	(event)	9 Participants recorded their interactions for 61-77 days ($M=65.56$, $SD=6.62$). The total number of social interactions reported per participant ranged from 114-360 ($M=248$, $SD=92.59$). Participants completed 0-16 ($M=3.91$, $SD=1.10$) questionnaires per day.
D5. An EMA study of interpersonal behaviour, affect, and perception in victims of bullying [45].	clinical psychology	(event)	For 14 consecutive days, each participant engaged in event-contingent recording of behaviours, perceptions, and affect in social interactions. In total 5606 reports were collected.
D6. Exploration of change processes in recurrent depression [51].	clinical psychology	(signal)	42 participants, sampled 10 times a day, 3 days a week, for 8 weeks. The study explored whether ESM can be used to detect within-individual affective change in remitted previously-depressed individuals, undergoing different relapse-prevention treatments.
D7. Study of anxiety in adolescents.	clinical psychology	(signal)	Approximately 90 participants sampled 5 times a day over the course of approximately 10 days. Approximately 3800 sessions gathered. Data still undergoing cleanup and analysis.
D8. In situ cognitive performance testing.	industrial psychology	(event)	Over the course of 2 weeks, 40 participants took cognitive tests 2 times a day, once in the morning and once in the afternoon, to test their alertness in relation to an evaluation of their sleep once in the morning.
D9. Towards task analysis tool support [33].	computer science	(signal)	12 participants over the course of 9 days and submitted 244 sessions. The study used EMA to support the analysis of collaborative and distributed tasks in an industrial setting.
D10. Personalised delivery of interventions for physiotherapy [35].	physiotherapy research	(signal)	36 participants for one week, were sampled 3 times a day. The goal was to compare participation compliance between an experimental and a control group.
D11. Study of television binge-watching behaviours [16].	media research	(event)	32 participants provided 132 data sessions, collected over the course of 10 days.
D12. Determinants of perceived sleep quality in normal sleepers [24].	clinical psychology	(event)	50 participants filled out a morning and an evening diary for two weeks. Eventually 1232 records were collected.
D13. Evaluation of interactions with an arduino-operated lighting system [40].	industrial design	(event)	5 people over a period of 4 weeks, received daily a questionnaire asking about their experience with the system.

Table 1. A list of studies that have used TEMPEST

Also, data-input timestamps helped the researchers ensure that the data was reported at the time of the event and prevent “back-filling”, a practice where participants will fill out a week’s worth of questionnaires on a single day. Finally, as pointed out by researchers, contrary to laying out sheets of paper to fill out, participants can inconspicuously respond to questionnaires without drawing attention to themselves, when in public. Interacting with a diary on the smartphone is perceived as one of many private interactions with one’s personal device.

5.1.2 Offline functionality. Offline recording of data is not possible with most online survey-distribution software. Offline functionality is important, as in EMA it is desirable that reports are provided as close to the event of interest as possible, and that should not be hindered by lack of Internet access at that time.

5.1.3 EMA as application component. In D1, D2 and D3, a motivation for using TEMPEST was that it removed the burden of building a surveying component for their own apps and decoupled concerns about their own software from the implementation of the survey. Compared to coding questions in their application code, researchers were able to forgo the burden of implementing a survey as such, and also be able to decide, edit and fine-tune the content of their survey items independently of their own programming efforts.

5.2 Requirements uncovered through the use of TEMPEST

During the course of several studies we were able to identify weaknesses of the system, or opportunities to extend it, which the reader could consider as extensions to the initial requirements for the composition of an EMA platform.

5.2.1 Allow other applications to initiate sampling sessions. In D1, D2, and D3 TEMPEST was part of other software systems, where the triggering event was automated by user interactions within other applications. In these cases, researchers used the TEMPEST GUI to compose their questionnaires and execution logic, while the initiation of the sampling session was performed by other programs. In D2 this would happen on Android by broadcasting Intents to the native TEMPEST client with information about which *Assignable Item* to instantiate. In D1 and D3, the remote app’s own WebView component would make HTTP requests to TEMPEST’s server to download its web-application client and render corresponding *Assignable Items*. In the future we will also allow HTTP requests from approved third party systems to trigger *Assignable Items* on TEMPEST’s own web and native clients. To this end, developers will only need to be aware of the service’s HTTP API, to make use of it. Devices in the Internet of Things will thus be able to initiate sampling sessions with their users, without a need for their own accompanying mobile app or infrastructure.

5.2.2 Allow fine control of presentation of Widgets. Most of the studies have used ordinary *Widgets* such as check-boxes and sliders. For D8 we extended the set of *Widgets* with interactive cognitive tests, using HTML5 Canvas. Additionally for D8, certain questionnaire items within a single *Screen* could become irrelevant given certain responses. To allow irrelevant items to be hidden, *Widgets* were extended to assert logical statements, also involving protocol variables, so as to determine individually whether they should appear within a given instance of a *Screen*. Also, to allow a *Widget* to influence the appearance of other *Widgets* of the same *Screen*, after it had been rendered, a value-change for the *Widget* will dispatch an event to its *Screen*, which will call on each of its member *Widgets* to re-evaluate their visibility.

5.2.3 Allow calculations on data collected to be performed as part of the protocol. Typically, researchers perform statistical calculations on data collected at a separate time, after the collection

of data has been completed, as part of a discrete data analysis phase. This phase will either involve a different module of the software suite used for the experiment, one that is solely concerned with the analysis, rather than the collection of data, or even a different software package altogether. D5 had been an iteration on the protocol followed by D4, both involving social encounters, and both shared a similar data analysis process. In D5 the researchers identified the opportunity to compute some of the desired statistical calculations by a programmed component of the protocol, as the protocol was executed. This resulted into a collected data set where part of the data-analytic effort had already been performed, easing the burden of later analysis.

5.2.4 *Allow EMA programs to incorporate custom scripting for more automation during the study.*

While the opportunity to perform statistical calculations at the time of protocol execution was identified in D5, so as to produce a richer initial dataset, the system did not provide pre-made functions to allow this. Instead, a quick implementation of such functions was possible without extensions to the system itself, by way of a scripting *Widget*. Scripting in TEMPEST allows the easy definition of custom *Widgets* behaviours, on the level of protocol composition. Scripting *Widgets* can read the TEMPEST runtime environment's memory for other *Widget* values, perform calculations, and report results as variable values of their own. Scripting *Widgets* are non-interactive and do not appear on the participant's screen. Future extensions will allow the scope of the EMA protocol, either through specific *Widgets* or specialised *Assignable Items*, to extend also to execution on the server, for access to the greater dataset across multiple participants.

5.2.5 *Offer warnings for unusual cases to avoid potential errors.* As a system offering an end-user programming model for researchers, TEMPEST should also provide safeguards and warnings against mistakes and omissions that they are likely to make. Early on, it became apparent that it is useful to provide namespacing for *Widget* variables based on the names of *Screens*. In this way, co-existing duplicates of *Screens* do not require researchers to rename every *Widget* that they contain. As a consequence, names of *Assignable Items* also became unique, and warnings were placed in the editing interface's text-fields for both for naming the *Assignable Items*, and for providing valid variable names (e.g. not containing special characters). These warning fields repeatedly check with the server for the uniqueness of the typed name, as the researcher types each letter. Additionally, while certain programming environments (e.g. JavaScript) do allow multiple declarations of variables with the same name, we found in most use cases that when this happened in TEMPEST, it would be due to oversight, rather than deliberate planning. A warning to the researcher that a newly declared variable will overwrite another would be useful.

5.2.6 *Allow the protocol to be easily inspected and understood.* In D4, D5, D6, D7, more researchers than our principle contact were involved in order to help with various aspects of the experiment's execution, such as managing participants. Many were not creators of the EMA program, but rather maintainers or re-implementers. The protocol, as expressed in the GUI interface, does not make all of its details immediately apparent to the reader, e.g., certain attributes are not visible in *Assignable Item* previews, but only during editing. Consequently, certain aspects of the programmed behaviour could escape a researcher's attention when reproducing a sequence of execution and could lead to misalignment with the purposes of the data collection. We realised that it is important to offer to researchers representations of the protocol that are readable, unambiguous and detailed enough so that they can be easily shared, modified and executed between different teams and experiments, and we have taken steps in this direction as well, by expressing EMA protocols as HTML documents.

5.3 Evaluation by participants

Our main focus has been to tailor the system to the needs of the researchers, under the assumption that they are a super-set of the needs of participants as users of the system. Researchers consider the participants' needs for ease of understanding and use of the system a priority and constantly test and pilot to ensure that the system is not only functional but also highly usable.

Unfortunately, it was not possible to collect feedback from the participants of all the deployment studies as we did not want to constrain/compromise the study design for each of the researchers by our own data collection. Further, any feedback by participants would not discriminate between the capabilities and the design of the platform, and the questionnaire design itself, so interpretation of such feedback would be challenging.

However in D4, part of the researchers' aims was to evaluate TEMPEST itself for administering the social interaction questionnaires, and we used the occasion to pursue an evaluation of the system by participants themselves. The System Usability Scale questionnaire [8] was distributed to the 9 participants in the study, to evaluate their experience with TEMPEST. SUS scores ranged from 75-95 ($M=84.17$, $SD=7.60$), suggesting that all participants had positive experiences with the software.

Some participants reported that occasionally they did not complete some questionnaires because the software was responding slowly, or because they forgot to take or charge their phone. However, participants evaluated their overall experience with the software as positive.

6 SUMMATIVE EVALUATION

This section discusses a summative evaluation that collected post-hoc data regarding the platform. It is comprised of two parts, administered as a combined questionnaire. The first part asked researchers about their goals and how they had been satisfied or not by the system (Table 2), and also asked them to provide comparisons with other systems (Table 3). The second part consisted of the Unified Theory of Acceptance and Use of Technology (UTAUT) questionnaire [56].

The questionnaire was set up and distributed via TEMPEST itself to principal researchers, who had come into contact with us and had used the system to conduct one or more studies with. We also asked them to forward the questionnaire to any collaborators they might have recruited. Eventually responses were collected from 18 individuals.

Asked about their roles in the studies, 13/18 (72%) respondents answered they had a supervisory role, 12/18 (67%) had a role in composing the questionnaires, 14/18 (78%) had a role in supervising the data collection and 11/18 (61%) supervised the cohort's participation.

6.1 Researcher goals

Three open questions asked "what were you hoping to achieve using TEMPEST?", "in what way did TEMPEST succeed in meeting your needs?", and "in what way did TEMPEST fail in meeting your needs?". Table 2 presents the coded responses and their frequencies.

Responses show that the researchers were in need of a versatile tool specifically built for intensive longitudinal methods, hoping for greater support than they had experienced with other materials. Some misfunctions were encountered, but overall the system was perceived as versatile and easy to use.

6.2 UTAUT survey

We asked participants to respond to the Unified Theory of Acceptance and Use of Technology (UTAUT) questionnaire [56]. In their original work, Venkatesh et al. derived the UTAUT questionnaire from eight different user acceptance models and hypothesised that seven constructs are

“What are/were you hoping to achieve with TEMPEST?” (frequency)	
intensive longitudinal study	(11)
improvement in materials or methodology	(3)
ease of use	(2)
“In what way did TEMPEST succeed in meeting your needs?” (frequency)	
ease of use	(6)
versatility in protocol composition	(6)
reliability	(3)
efficiency	(2)
monitor study in realtime	(1)
“In what way did TEMPEST fail in meeting your needs?” (frequency)	
ease of data processing	(6)
system misfuctions	(3)
missing features	(2)

Table 2. Coded user feedback on what they were hoping to achieve, what needs were met and what needs were not met, with frequencies of mentions.

“How does the system compare with other systems?”	
“I actually do not know. I know one other system which I used years ago and that system had less freedom than this one which was annoying.”	
“Haven’t got experience with other systems which could do the same.”	
“I think it is nice that it works on their own phone (as apposed to the psymate). The interface is not very attractive, which I think doesn’t help compliance. But most importantly, participants were very annoyed (irritated) when there were difficulties with the app. It very much helped that data was uploaded immediately, so that we (the researchers) could monitor compliance and provide feedback.”	
“I don’t know any similar system.”	
“Nicer interface for the participants, compared to google forms for instance, the opposite for the researcher.”	
“Have never used other systems, but the compatibility with mobile phones is a definite plus.”	
“The option to fill it in offline is a great tool, as this made the compliance very high. Especially for older populations, as they are not always that skilled to use the internet or computer.”	
“[TEMPEST is the] only available system in my opinion to collect field data of cognitive tests on a smartphone (no comparison in that sense possible); data output is complex compared to online computer systems; very easy to use offline after installing and easy to set up a survey; so very convenient for ambulatory studies. The only comparison I can make is with a PDA, [TEMPEST] was much more convenient.”	

Table 3. Answers provided to the question “How does the system compare to other systems?”

direct determinants of a user’s behavioural intention to use a software system. These constructs are performance expectancy, effort expectancy, attitude toward using technology, social influence, facilitating conditions, self-efficacy, and anxiety. They are rated against a 7-point Likert scale.

Item	mean	SD
Performance Expectancy ($\alpha=0.72$)	5.611	0.981
U6: I would find the system useful in my job.	5.889	0.9
RA1: Using the system enables me to accomplish tasks more quickly.	5.667	0.970
RA5: the system increases my productivity.	5.278	1.074
Effort Expectancy ($\alpha=0.89$)	5.249	1.007
EOU3: My interaction with the system would be clear and understandable.	5.111	1.079
EOU5: It would be easy for me to become skilful at using the system.	5.667	0.970
EOU6: I would find the system easy to use.	5.165	0.924
EU4: Learning to operate the system is easy for me.	5.056	1.056
Social Influence ($\alpha=0.72$)	4.472	1.581
SN1: People who influence my behaviour think that I should use the system.	4.167	1.757
SN2: People who are important to me think that I should use the system.	4.167	1.618
SF2: The senior management of this institution has been helpful in the use of the system.	4.444	1.542
SF4: In general the organisation has supported the use of the system.	5.111	1.410
Attitude Toward Using Technology($\alpha=0.82$)	5.278	0.946
A1: Using the system is a bad idea.	1.944	0.873
AF1: The system makes work more interesting.	5.167	1.043
AF2: Working with the system is fun.	4.722	0.826
Affect1: I like working with the system.	5.167	1.043
Facilitating Conditions($\alpha=0.5$)	5.445	1.323
PBC2: I have the resources necessary to use the system.	5.167	1.098
PBC3: I have the knowledge necessary to use the system.	5.556	1.042
PBC5: The system is not compatible with other systems I use.	2.833	1.339
FC3: A specific person (or group) is available for assistance with system difficulties.	5.889	1.811
Self Efficacy ($\alpha=0.29$)	4.750	1.443
I could complete a job or task using the system—SE1: ...if there was no one around to tell me what to do as I go.	4.556	1.653
SE4: ...if I could call someone for help if I got stuck.	6	1.029
SE6: ...if I had a lot of time to complete the job for which the software was provided.	4.556	1.381
SE7: ...if I had just the built-in help facility for assistance.	3.889	1.711
Anxiety ($\alpha=0.73$)	3.028	1.416
ANX1: I feel apprehensive about using the system.	3.667	1.609
ANX2: It scares me to think that I could lose a lot of information using the system by hitting the wrong key.	3.167	1.581
ANX3: I hesitate to use the system for fear of making mistakes I cannot correct.	2.556	1.294
ANX4: The system is somewhat intimidating to me.	2.722	1.179

Table 4. Responses to the UTAUT questionnaire. Chronbach's alpha values are also reported for each scale. Most items present high consistency, with the exceptions of self-efficacy and facilitating conditions.

The UTAUT questionnaire is of interest also because it elucidates aspects of a system and its use, that can help identify strengths and weaknesses in its implementation. These relate to the clarity of its presentation, its facilities to support learning its operation or enable users to perform troubleshooting. We take a qualitative view of the responses and use the questionnaire as a guide in deciding where to pay attention in developing the system further. Lower scores in questionnaire items might point to weaknesses that further iterations on the development can prioritise.

Table 4 presents the responses of our 18 respondents. Their reactions were positive across most items in all scales. Attitude Towards Technology indicates a positive overall affective reaction towards using the system, but certain aspects of the system could be made more accessible to non-expert users. In Social Influence, responses reflect the fact that the decision to use the system has been mostly that of the users themselves, and that the organisation (a university department in most cases) was not involved in the deployments. Scores for Anxiety were relatively low. The task of running an EMA study is not one of typical software use. Typically, users operate a software application directly, in local settings. On the contrary, EMA studies involve running a distributed system, operating on hardware and OS configurations that might vary widely, with multiple users in remote settings, who are requested to invest time, and they often experience fatigue because of the sampling process. As a result, misfunctions can be costly, and troubleshooting can be arduous.

Low internal consistency for Self Efficacy and Facilitating Conditions could be theorised to reflect the differences in support that could be provided to researchers of different skill levels at any given time. The different nature of event-contingent versus signal-contingent recording in different studies could also be a contributing factor.

7 SUMMARY

In summary, TEMPEST was perceived to be a useful system that provides substantial benefits over paper-based methods or generic survey software not built for the task.

It was also perceived as easy to use. However, for respondents who come from a background of clinical psychology, the conceptual model of EMA as a program was not always intuitive and the terminology used by the system (*Conditionals*, *Sequentials*, *Assignable Items*) was not always immediately clear.

The system proved effective in that researchers were able to carry out the study as end-user developers of their own EMA protocol. TEMPEST was sufficiently powerful to support all the studies above, and allowed protocol elements to be adapted and reused between cases.

Although the system operated reliably enough for the studies under examination to be conducted, some bugs were nonetheless uncovered during the execution of the studies (and were corrected). Interestingly however, in some cases where researchers would report a system malfunction as a 'bug' were in fact errors in the protocol itself, showing that end-user software development of EMA protocols also needs to be supported by tools to support 'debugging', as programmatic errors can be introduced at that level too.

Future improvements should expand on prevention and handling of errors in the protocol, providing more guidance to the novice user of TEMPEST (documentation and help screens to introduce them to the conceptual model) and extend the conceptual model and system functionality to support monitoring compliance and participant status as the study unfolds. Data analysis was beyond the scope of the system, but feedback indicates that options for data processing could be considered.

8 CONCLUSION

EMA is an approach to data collection that requires collection of data over sustained periods, and in the diverse locations people find themselves as they go about their daily activities. This approach

has largely benefited from mobile Internet access through smartphones, and how to build software in its support becomes more interesting for research, as the availability, computational capabilities and connectivity of the personal devices increase.

We have elaborated on a paradigm for the organisation of EMA systems as programmable platforms for the purpose of in situ data collection. We also presented TEMPEST, a platform that supports domain experts in setting up their own EMA studies, following an end-user development approach. In earlier works [4][3] we have described only the architectural principles that would allow platform independent execution of the EMA protocols and that would solve the challenges of synchronising data between handheld client and the back end system in a manner transparent to the user. The system presented here is complete with structures that enable the composition of EMA programs by end-users. To our knowledge, it is the first implemented, and extensively tested in real use system, to offer a browser-based runtime environment that conceptualises the composition of EMA protocols as programs, the construction of which is supported by a GUI, at a level of abstraction relevant to the concerns of researchers.

Specifically, the advance reported is in support of a mental model of the EMA protocol as a program executed on a virtual machine. This virtual machine abstracts away from the software architecture to consider the combined behaviour of the human-respondent and the computing infrastructure, aiming to reduce that gap that a researcher has to bridge to articulate their conception of a study protocol in a way that can be interpreted and executed by a computational platform. Relevant details of the implementation have also been provided.

Finally the paper has contributed evidence of the extensive use of the platform in real life projects: the size and realism of the research studies, many of which have been published, indicates the adequacy and pertinence of the proposed approach. By surveying attitudes of the researchers who have used the platform in these studies, regarding technology acceptance, we could show that they would use the platform again. Despite them being non-technical experts and the fact that this technology is only supported by the researcher who developed it, there was a high level of acceptance reported, and confidence by the researchers that they can use TEMPEST for setting up and administering their studies.

Future work will focus on sharing and reuse of EMA protocols as programs. It will expand on the topic of EMA program expression in textual format, which can be easy to comprehend, modify, share online and in print, and at the same time remain executable on platforms that adopt the proposed paradigm.

REFERENCES

- [1] N. Aharony, A. Gardner, C. Sumter, and A. Pentland. Funf: Open sensing framework, 2011.
- [2] D. J. Barrett and D. Barrett. Esp, the experience sampling program. Retrieved March, 1:2007, 2005.
- [3] N. Batalas and P. Markopoulos. Considerations for computerized in situ data collection platforms. In *Proceedings of the 4th ACM SIGCHI symposium on Engineering interactive computing systems*, pages 231–236. ACM, 2012.
- [4] N. Batalas and P. Markopoulos. Introducing tempest, a modular platform for in situ data collection. In *Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense Through Design*, pages 781–782. ACM, 2012.
- [5] N. Batalas, J. Quevedo-Fernandez, J.-B. Martens, and P. Markopoulos. Meet your users in situ data collection from within apps in large-scale deployments. *International Journal of Handheld Computing Research (IJHCR)*, 6(3):17–32, 2015.
- [6] F. M. Bos, R. A. Schoevers, and M. aan het Rot. Experience sampling and ecological momentary assessment studies in psychopharmacology: A systematic review. *European Neuropsychopharmacology*, 25(11):1853–1864, 2015.
- [7] R. Bosman, C. Albers, N. Batalas, and M. aan het Rot. No cyclicality in mood and social functioning in women with self-reported pms. Manuscript submitted for publication.
- [8] J. Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.
- [9] M. Capatu, G. Regal, J. Schrammel, E. Mattheiss, M. Kramer, N. Batalas, and M. Tscheligi. Capturing mobile experiences: Context-and time-triggered in-situ questionnaires on a smartphone. *Measuring Behavior 2014*, 2014.

- [10] S. Carter, J. Mankoff, S. R. Klemmer, and T. Matthews. Exiting the cleanroom: On ecological validity and ubiquitous computing. *Human-Computer Interaction*, 23(1):47–99, 2008.
- [11] T. C. Christensen, L. F. Barrett, E. Bliss-Moreau, K. Lebo, and C. Kaschub. A practical guide to experience-sampling procedures. *Journal of Happiness Studies*, 4(1):53–78, 2003.
- [12] T. S. Conner and M. R. Mehl. Ambulatory assessment: Methods for studying everyday life. *Emerging Trends in the Social and Behavioral Sciences: An Interdisciplinary, Searchable, and Linkable Resource*, 2015.
- [13] S. Consolvo, B. Harrison, I. Smith, M. Y. Chen, K. Everitt, J. Froehlich, and J. A. Landay. Conducting in situ evaluations for and with ubiquitous computing technologies. *International Journal of Human-Computer Interaction*, 22(1-2):103–118, 2007.
- [14] P. Copley. Research and evaluation of marketing communications. In *Marketing communications management: concepts and theories, cases and practices*, chapter 18. Routledge, 2004.
- [15] M. Csikszentmihalyi, R. Larson, and S. Prescott. The ecology of adolescent activity and experience. *Journal of youth and adolescence*, 6(3):281–294, 1977.
- [16] D. de Feijter, V.-J. Khan, and M. van Gisbergen. Confessions of a ‘guilty’ couch potato understanding and using context to optimize binge-watching behavior. In *Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video*, pages 59–67. ACM, 2016.
- [17] D. Estrin, M. Carroll, and N. Lakin. Researchstack. <http://researchstack.org>. Accessed: 2017-01-02.
- [18] B. EVANS. Applying computational methods to scale qualitative methods. In *Ethnographic Praxis in Industry Conference Proceedings*, volume 2016, pages 348–368. Wiley Online Library, 2016.
- [19] D. Ferreira, V. Kostakos, and A. K. Dey. Aware: mobile context instrumentation framework. *Frontiers in ICT*, 2:6, 2015.
- [20] J. Firth and J. Torous. Smartphone apps for schizophrenia: a systematic review. *JMIR mHealth and uHealth*, 3(4):e102, 2015.
- [21] J. E. Fischer. Experience-sampling tools: a critical review. *Mobile living labs*, 9:1–3, 2009.
- [22] R. C. Fraley and N. W. Hudson. Review of intensive longitudinal methods: An introduction to diary and experience sampling research, 2014.
- [23] 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. In *Proceedings of the 5th international conference on Mobile systems, applications and services*, pages 57–70. ACM, 2007.
- [24] M. Goelema, M. Regis, R. Haakma, E. van den Heuvel, P. Markopoulos, and S. Overeem. Determinants of perceived sleep quality in normal sleepers (in press). *Behavioural Sleep Medicine*, 2017.
- [25] A. Gordon. SurveyMonkey. com—A web-based survey and evaluation system: <http://www.surveymonkey.com>, 2002.
- [26] J. M. Hektner, J. A. Schmidt, and M. Csikszentmihalyi. *Experience sampling method: Measuring the quality of everyday life*. Sage, 2007.
- [27] T. Hendela. Apple introduces researchkit, giving medical researchers the tools to revolutionize medical studies. [URL:http://www.apple.com/ca/pr/library/2015/03/](http://www.apple.com/ca/pr/library/2015/03/), March 2015. Accessed: 2016-12-29.
- [28] J. R. Hinrichs. Communications activity of industrial research personnel. *Personnel Psychology*, 17(2):193–206, 1964.
- [29] J. B. Hoeksma, S. M. Sep, F. C. Vester, P. F. Groot, R. Sijmons, and J. De Vries. The electronic mood device: Design, construction, and application. *Behavior Research Methods, Instruments, & Computers*, 32(2):322–326, 2000.
- [30] A. E. Hühn, V.-J. Khan, P. Ketelaar, J. van’t Riet, R. König, E. Rozendaal, N. Batalas, and P. Markopoulos. Does location congruence matter? a field study on the effects of location-based advertising on perceived ad intrusiveness, relevance & value. *Computers in Human Behavior*, 2017.
- [31] S. S. Intille, J. Rondoni, C. Kukla, I. Ancona, and L. Bao. A context-aware experience sampling tool. In *CHI’03 extended abstracts on Human factors in computing systems*, pages 972–973. ACM, 2003.
- [32] V. Javed Khan, P. Markopoulos, D. Dolech, A. Eindhoven, B. Eggen, and D. Dolech. Features for the future experience sampling tool, 2009.
- [33] S. Kieffer, N. Batalas, and P. Markopoulos. Towards task analysis tool support. In *Proceedings of the 26th Australian Computer-Human Interaction Conference on Designing Futures: the Future of Design*, pages 59–68. ACM, 2014.
- [34] J. Maloney, M. Resnick, N. Rusk, B. Silverman, and E. Eastmond. The scratch programming language and environment. *ACM Transactions on Computing Education (TOCE)*, 10(4):16, 2010.
- [35] P. Markopoulos, N. Batalas, and A. Timmermans. On the use of personalization to enhance compliance in experience sampling. In *Proceedings of the European Conference on Cognitive Ergonomics 2015*, page 15. ACM, 2015.
- [36] G. Metaxas, P. Markopoulos, and E. H. Aarts. Modelling social translucency in mediated environments. *Universal Access in the Information Society*, 11(3):311–321, 2012.
- [37] J.-K. Min, A. Doryab, J. Wiese, S. Amini, J. Zimmerman, and J. I. Hong. Toss’n’turn: smartphone as sleep and sleep quality detector. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 477–486. ACM, 2014.

- [38] D. S. Moskowitz and S. N. Young. Ecological momentary assessment: what it is and why it is a method of the future in clinical psychopharmacology. *Journal of psychiatry & neuroscience: JPN*, 31(1):13, 2006.
- [39] I. Myin-Germeys, M. Birchwood, and T. Kwapiil. From environment to therapy in psychosis: a real-world momentary assessment approach. *Schizophrenia bulletin*, 37(2):244–247, 2011.
- [40] S. Offermans. *Interacting with Light*. Eindhoven University of Technology, April 2016.
- [41] V. O'Reilly-Shah and S. Mackey. Survalytics: An open-source cloud-integrated experience sampling, survey, and analytics and metadata collection module for android operating system apps. *JMIR mHealth and uHealth*, 4(2):e46, 2016.
- [42] F. Paternò. End user development: Survey of an emerging field for empowering people. *ISRN Software Engineering*, 2013, 2013.
- [43] J. Quevedo-Fernández and J.-B. Martens. idanimate—supporting conceptual design with animation-sketching. In *Collaboration in Creative Design*, pages 251–270. Springer, 2016.
- [44] N. Ramanathan, F. Alquaddoomi, H. Falaki, D. George, C.-K. Hsieh, J. Jenkins, C. Ketcham, B. Longstaff, J. Ooms, J. Selsky, et al. Ohmage: an open mobile system for activity and experience sampling. In *2012 6th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*, pages 203–204. IEEE, 2012.
- [45] M. ranzen, G. Sadikaj, D. Moskowitz, B. Ostafin, and M. aan het Rot. Intra- and interindividual variability in the behavioural, affective, and perceptual effects of alcohol consumption in a social context. Manuscript submitted for publication.
- [46] C. Reynoldson, C. Stones, M. Allsop, P. Gardner, M. I. Bennett, S. J. Closs, R. Jones, and P. Knapp. Assessing the quality and usability of smartphone apps for pain self-management. *Pain Medicine*, 15(6):898–909, 2014.
- [47] D. Rough and A. Quigley. Jeeves—a visual programming environment for mobile experience sampling. In *Visual Languages and Human-Centric Computing (VL/HCC), 2015 IEEE Symposium on*, pages 121–129. IEEE, 2015.
- [48] J. D. Runyan, T. A. Steenbergh, C. Bainbridge, D. A. Daugherty, L. Oke, and B. N. Fry. A smartphone ecological momentary assessment/intervention app for collecting real-time data and promoting self-awareness. *PLoS One*, 8(8):e71325, 2013.
- [49] C. Schmitz et al. Limesurvey: An open source survey tool. *LimeSurvey Project Hamburg, Germany*. URL <http://www.limesurvey.org>, 2012.
- [50] S. M. Schueller, M. Begale, F. J. Penedo, and D. C. Mohr. Purple: a modular system for developing and deploying behavioral intervention technologies. *Journal of medical Internet research*, 16(7):e181, 2014.
- [51] C. Slofstra, M. Nauta, E. Holmes, E. Bos, M. Wichers, N. Batalas, N. Klein, and C. Bockting. Exploring the relation between visual mental imagery and affect in the daily life of previously depressed and never depressed individuals. *Cognition and Emotion*, 2017. In press.
- [52] A. A. Stone, C. A. Bachrach, J. B. Jobe, H. S. Kurtzman, and V. S. Cain. *The science of self-report: Implications for research and practice*. Psychology Press, 1999.
- [53] A. A. Stone and S. Shiffman. Ecological momentary assessment (ema) in behavioral medicine. *Annals of Behavioral Medicine*, 1994.
- [54] P. Totterdell and S. Folkard. In situ repeated measures of affect and cognitive performance facilitated by use of a hand-held computer. *Behavior Research Methods, Instruments, & Computers*, 24(4):545–553, 1992.
- [55] D. Trossen and D. Pavel. Aircs: A mobile sensing platform for lifestyle management research and applications. In *International Conference on Mobile Wireless Middleware, Operating Systems, and Applications*, pages 1–15. Springer, 2012.
- [56] V. Venkatesh, M. G. Morris, G. B. Davis, and F. D. Davis. User acceptance of information technology: Toward a unified view. *MIS quarterly*, pages 425–478, 2003.
- [57] P. Wilhelm and M. Perrez. A history of research conducted in daily life (scientific report nr. 170). 2013.
- [58] M. Wolf-Meyer. Therapy, remedy, cure: disorder and the spatiotemporality of medicine and everyday life. *Medical anthropology*, 33(2):144–159, 2014.
- [59] H. Xiong, Y. Huang, L. E. Barnes, and M. S. Gerber. Sensus: a cross-platform, general-purpose system for mobile crowdsensing in human-subject studies. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 415–426. ACM, 2016.