SGVis: Analysis of Data From Mass Participation Ubicomp Trials

Alistair Morrison, Matthew Chalmers
School of Computing Science, University of Glasgow, UK
{alistair.morrison,matthew.chalmers}@glasgow.ac.uk

ABSTRACT
The recent rise in popularity of ‘app store’ markets on a number of different mobile platforms has provided a means for researchers to run worldwide trials of ubiquitous computing (ubicomp) applications with very large numbers of users. This opportunity raises challenges, however, as more traditional methods of running trials and gathering data for analysis might be infeasible or fail to scale up to a large, globally-spread user base. Here we present SGVis, a data analysis tool designed to aid ubicomp researchers in conducting trials in this manner. This paper discusses the difficulties involved in running large scale trials, explaining how these led to recommendations on what data researchers should log, and to design choices made in SGVis. We describe the SGVis tool in depth, outlining several methods of use and why they help with challenges raised by large scale research. A means of categorising users is also described that could aid in data analysis and management of a trial with very large numbers of participants. SGVis has been used in evaluating several mass-participation trials, involving tens of thousands of users, and we describe several use cases that describe its utility.

Keywords: Mass participation, ubicomp, data analysis, visualisation, iPhone, iOS, categorisation

INTRODUCTION
The rise of ‘app store’ markets is a relatively recent phenomenon. The Apple App Store launched in July 2008 and saw its 10 billionth download in January 2011 (Reuters, 2011). Smartphone usage has also seen a sharp rise in usage in recent years, with market research firm IDC suggesting that 15.4% of the mobile phone market consisted of smartphones at the beginning of 2010 (Llamas, 2010), and several different mobile platforms now offer ‘app store’ software distribution mechanisms. The combination of this growing potential user base and popular online software repositories provides a relatively simple way to recruit users for worldwide trials of ubiquitous computing (ubicomp) applications and several researchers are beginning to use this ‘mass participation’ approach (McMillan et al., 2010; Cramer et al, 2010).

Distributing ubicomp trial applications in this way provides great opportunities for researchers in terms of potentially reaching a very large number of trial participants. For example, Hungry Yoshi, a game discussed below, has had over 40,000 users in the 12 months since release.

The potential advantages of deploying trial software to a wide audience are numerous. These include getting a larger sample size to provide more certainty to quantitative analyses and, additionally, a global release of software provides the opportunity to reach users from vastly different geographic locations so as to help reduce cultural biases stemming from recruitment of only locally-based participants.
The benefits to be gained from this style of deployment do come with some potential drawbacks, however, as running a trial with a large number of users over a vast geographical area can raise significant challenges. Compared to more traditional local deployments of software, researchers can be further removed from the trial, unable to meet participants and perhaps less able to closely observe the use of the software under examination. Additionally, having such a large user base could lead to an overwhelming amount of data being generated and researchers might not have the resources to study every user in detail.

We have designed several tools to aid researchers in conducting such trials. The previously published SGLog framework (Hall et al., 2009) provides a simple means for developers to instrument their mobile application code and stream log data back to researchers’ database. This paper presents SGVis - a complementary desktop analysis tool designed to allow evaluators to study this data. Several modes are available in SGVis, allowing data gathered from users’ devices to be processed in various ways. In one mode an individual’s use of trial software can be studied in detail, while in another overall trends for an application can be analysed, such as download patterns over time or the average number of times participants perform certain actions. A ‘live’ mode allows analysts to watch software use ‘as it happens’, with a collection of maps and summary statistics showing researchers which of their trial applications are being used anywhere in the world in the last few minutes. The final mode in SGVis allows analysts to categorise the many thousands of users they might have, using multidimensional data analysis, clustering and visualisation techniques. In this way, analysts might be able to more successfully manage categories of participants rather than attempting to study each of the thousands of logged users in detail. SGVis has been used in the evaluation of several mobile applications, handling data sets of tens of thousands of users spread worldwide.

The following section surveys related work in the area of mass participation research applications, and of analysis tools that can be used to study data captured from such applications. A description is provided of the Contextual Software project (UK EPSRC EP/F035586/1), and how SGVis forms part of this research. This is followed by a discussion of the challenges arising from conducting research in this manner and recommendations for the types of data other researchers working in this area might want to capture. Thereafter, the SGVis tools for analysing large scale ubiquitous computing trials are introduced, including a rationale for design and illustrations of their utility with specific use cases, before finally a discussion is presented of the use of SGVis and ongoing work in this area.

RELATED WORK
Researchers in ubiquitous computing have recently begun to take advantage of ‘mass participation’ methods of distributing software, with several research applications being released via public software repositories.

An early landmark of large-scale deployment of ubicomp applications that pre-dated ‘app store’ style software repositories was Mogi Mogi. As reported by Licoppe and Inada (2006), this location-based mobile multiplayer game was released commercially in Japan, and in 2004 had roughly 1,000 active players. One of the earliest research systems to be released on Apple’s App Store is CenceMe, which uses context sensing to automatically updates social networking sites with a user’s current activity (Miluzzo, Lane, Lu & Campbell, 2010). CenceMe was originally developed for the Nokia N95, but re-implemented for the iOS platform in time for the App Store launch in July 2008. Hungry Yoshi, another iOS-based application released in September 2009, is a game that uses detected Wi-Fi access points as a
game resource, thus exposing underlying infrastructure through a game mechanic. In (McMillan et al., 2010) we show that it is possible to conduct both quantitative and qualitative research with a large, global user base.

A number of research projects have also been released through the Android Market. For example, My2cents allows users to scan barcodes of retail products and discuss them with other users (Karpischek & Michahelles, 2010). WorldCupinion (Rohs, Kratz, Schleicher, Sahami & Schmidt, 2010), an app for sharing real-time feedback during football matches, was also released on the Android Market and gained 1,645 downloads during the World Cup. Another example, AppAware, is a location-aware application that tracks installs and uninstalls of other software, to inform users of which apps are currently popular among other Android users in the local vicinity. The authors report that AppAware has had over 19,000 unique users, of whom 9,500 were actively using the application at the time the authors wrote the paper (Girardello & Michahelles, 2010).

In many of these studies, reported analysis so far is limited to download numbers and brief reports of usage statistics, often as provided by the software repository. Our own early experience of studying data from mass participation applications was often limited to raw SQL queries, or custom-built tools to handle the data specific to each iOS application. This obviously required considerable time and effort in each case, and the created systems might be tightly coupled to the evaluated system, offering limited potential for re-use.

The goal of SGVis is to create a general analysis toolkit that is reusable across studies of many mobile applications. Commercial organisations such as Flurry offer a similar service, providing developers with a logging framework and displaying online statistics on application usage (Flurry, 2010). However, Flurry’s tools only offer aggregate data, and our work with SGLog and SGVis goes further in allowing individual users to be studied in detail and allowing more complex forms of statistical analysis.

Other work in visualising data generated from ubicomp research projects focuses on analysing sensor data. For example, the Cityware system (O’Neill et al., 2006) recorded Bluetooth data and presented a chart of devices, based on aggregate densities and flows of people in particular urban areas. Similar work has looked at coarse-grained city-scale maps of people’s density based on concentrations of mobile phone signals sampled from GSM infrastructure (Reades, Calabrese, Sevtsuk & Ratti, 2007), and scans of Bluetooth data from crowds of people attending sports stadia (Morrison, Bell & Chalmers, 2009).

Other researchers have explored ‘experience sampling’ methods, in which a questionnaire appears on-screen when the mobile device detects that it is in a context of interest (Froehlich, Chen, Smith & Potter, 2006). Carter, Mankoff and Heer (2007) developed Momento, which supports experience sampling, diary studies, capture of photos and sounds, and messaging from evaluators to participants. It uses SMS and MMS to send data between a participant’s mobile device and an evaluator’s desktop client. Systems such as Replayer (Morrison, Tennent & Chalmers, 2006; Morrison, Tennent, Chalmers & Williamson, 2007) and DRS (Greenhalgh, French, Tennent & Humble, 2007) have been used for smaller-scale studies, combining data logged on participants’ mobile devices with video recorded by trial evaluators in the field. Events detected in system logs could then be used to filter video recorded by roaming evaluators or cameras mounted in fixed locations, thereby easing the process of finding relevant video data from potentially many hours of recorded footage. Although such systems can allow researchers to study users in greater numbers and at larger geographic and time scales than they can directly observe,
this approach does not scale up to public software repository-style releases, where analysts are unlikely
to gain video footage of participants using the system.

**Contextual Software Project**

The work presented in this paper is part of the Contextual Software project, which aims to provide tools and practices for developers and evaluators, and so improve the process of creating software that will have sustained contextual fit. The project has several components, such as a logging framework, a mechanism for delivering updates to deployed software and several ‘app store’-released applications. These are briefly described here, although the focus of this paper is the SGVis desktop analysis tool.

Several ubicomp applications have been developed for Apple’s iOS platform (meaning they can run on the iPhone, iPod Touch and iPad) as part of this project. Two iOS applications in particular are used to provide examples in this paper. Hungry Yoshi, a game that uses Wi-Fi infrastructure as a game resource, was the first application we released through the app-store method. It has had around 200,000 downloads and currently has around 40,000 registered users. The game is described by McMillan et al. (2010), where it was evaluated using a basic set of tools that would later evolve into the SGVis toolkit described here. Our most recent application is World Cup Predictor, a game designed to run alongside the FIFA World Cup, which tried to encourage social interaction with other players through Bluetooth-based data transfers. Evaluation of this trial is still ongoing, based around data analysis in SGVis. Both apps were free to download via an APT-based repository (McMillan, 2010).

SGVis displays data gathered via the complementary SGLog framework (Hall et al., 2009). SGLog provides a simple means to instrument iOS applications with timestamped general or application-specific event logging. An application equipped with SGLog creates local caches of log data on the device while the application is running, and this data is opportunistically uploaded to a server when the device has internet connection. A more traditional ubicomp study might recruit locally-based users, whose devices would log usage data to text files. Devices might be collected at the end of a trial for log data to be retrieved and data analysis to commence. Using an app store distribution method, researchers cannot physically gather the devices, so the SGLog framework captures log data during a trial and regularly uploads these logs to our server over a device’s Wi-Fi or cellular data connection. Such a system allows researchers access to more or less live data, eliminating the necessity to wait for devices to be returned before analysis can commence.

The SGLog infrastructure is explained in detail by Hall et al. (2009), along with our mechanism for allowing dynamic updates of deployed applications. Here we focus on the SGVis data analysis tools, designed for use with SGLog.

**LARGE SCALE TRIALS: WHAT DATA TO LOG?**

If other researchers wish to run trials in this manner, what types of data should they capture for analysis? A simple answer might be that researchers would like to receive as much data as possible. One of the features characterising research projects that employ app store-style distribution methods is the potential for long-duration trials. If research questions might evolve over the course of such a study, researchers will not be certain in advance of all the data they will require. In such circumstances it would be safer to have logged extra data and not require all of it than to have omitted to log a crucial element. Deciding upon what data to capture might therefore be equivalent to what *can* be logged, dictated by what is technically feasible on the platform.
It was described in the previous section how activity is logged on mobile devices and these logs are uploaded back to the researchers’ database. Were this transmission to take place over cellular data networks, participants might incur charges from their network providers, depending on the type of contract they have. If the participant was using the device abroad, this charge could be particularly high. As a workaround to this, the application can detect the current type of connection and the logging software can be configured to only upload data when users are connected to the Internet over WiFi. If it was detected that the application was on a cellular network rather than WiFi connection, the application could be configured to only upload what researchers deem the most ‘important’ log data. Researchers might view this as a tradeoff, balancing the amount of information they upload with the danger of annoying the users to the extent that they will stop using the application.

**Ethics and Privacy Concerns**

Of course, considerations on this tradeoff must also be balanced by researchers’ ethical responsibilities. An application should make clear any data that is being uploaded and allow users to make a decision on whether the use of the software is worth any potential monetary cost. Further ethical issues extend beyond purely financial considerations. As discussed above, it might be of benefit to researchers to acquire as much as data as possible, but is it appropriate to log vast amounts of data on users, even information that may be irrelevant to the study? We are addressing some of these concerns in ongoing work.

All applications that use SGLog present a privacy statement on first launch, explaining all the data that will be collected. Users must agree to this before they can use the application. Having done so, uploaded data from each user is timestamped and stored on a secured database on a central server. To protect the privacy of participants, this framework uses TLS to encrypt data sent between mobile devices and the server.

As releasing software via an app store means reaching a potentially global audience, it can not be assumed that all users will be fluent in English. This information is therefore presented in at least four different languages. A contact email address is also supplied for users to opt out of the trial at any time, and it is explained that all data collected from a user will be destroyed on receipt of such a request.

SGLog does not directly address repositories’ varied policies as to what one can log and programmers must understand what is acceptable in the repository they use for distribution.

**General and application-specific logging**

The data that researchers might choose to log can generally be divided into general or application-specific items. Data on location, accelerometer readings or battery levels is generic information that might be useful to record in all applications. It is more difficult to give general advice for application-specific logging, as each specific trial will obviously have a different focus and require different data. In general, we have found that recording screen changes and button clicks provides a useful overview of the usage of trial software.

The particular challenges of running large-scale trials mean that some forms of data might be viewed as being more crucial than others. For example, as researchers do not meet their participants at any point during the trial, it can be important to gain some basic information in order to compile demographic statistics and to understand the composition of the user base. Some forms of demographic information cannot be gathered automatically and would have to be asked of users: age and occupation, for example.
Device demographics, such as device model or OS version can be automatically logged. The following sections show examples of these type of demographic spreads.

It is impossible to provide a definitive list of the information necessary to run a large scale trial, just as it is impossible to create an analysis tool that is certain to fully answer all questions researchers will have in every future project. We hope we have provided some general pointers towards the set of information that will be of generic use. In the following sections, we describe the SGVis tool which, used in conjunction with this logged data, aims to tackle some of the difficulties incurred when running trials on such a scale.

**ANALYSIS TOOLKIT DESIGN CHOICES**

As well as the previously noted challenges, large scale trials do offer huge benefits in the ability to gain large amounts of participants, so SGVis has been specifically designed to harness the advantages of large numbers for quantitative analysis, calculating and displaying trends across the large numbers of users.

In the first days following the release of Hungry Yoshi we felt detached from the ongoing trial, and unable to closely follow the game’s use. Limited to running awkward SQL queries on the database, there was no simple means of monitoring the application’s usage and maintaining ‘peace of mind’ that everything was running smoothly. We elected to build a ‘live’ view of the trial to overcome researcher detachment in this way via a set of monitoring and visualisation tools that collect and visualise data in close to real-time. This allows researchers to keep in touch with an ongoing trial, and, as described in an example below, to identify problems as they arise.

Another strategy we adopted early on in the trialling of app store-released software was the use of Flurry (Flurry, 2010) to collect log data and see general patterns of usage. However, it soon became apparent that the overviews provided were insufficient to enable in-depth views of user activity. Only in studying individual users in detail could we select interesting people to interview and to prepare questions for such interviews based on observed behaviour. Therefore, as described in the following sections, SGVis provides analysts with the means to view aggregate data from all users, yet still be able to study an individual participant’s activity, and easily move between these views.

Having such a large number of users means it might not be practical to study each individual in detail. We have found that our applications attract tens of thousands of users. Therefore we also describe a mechanism to categorise users based on their recorded usage of the application, via a series of data processing, clustering and visualisation techniques. We suggest ways in which this categorisation might be used in analysis.

**SGVIS ANALYSIS TOOLS**

Our development work in mobile applications has concentrated on the iOS platform for our large distribution trials, and the SGVis desktop tool is a Cocoa application written in Objective-C. A server backend uses PHP to communicate with SGLog’s MySQL database. Hall et al (2009) describe how the SGLog framework can be added simply to an iOS project, with a single line of code necessary for each event researchers wish to log. SGVis setup is then simply a matter of pointing SGVis towards an SGLog database. The software will automatically configure itself and populate its views with data from all the different trials logged to that database. SGVis has four main modes: viewing quantitative data aggregated by user or app, ‘live’ mode to monitor current usage, and analysis of derived statistical data. The following sections will describe how each of these aims to facilitate the running and evaluation of large scale ubicomp trials.
**Trial data viewed by application**

The first feature of SGVis to be described is analysing aggregations of the potentially vast amounts of data generated by a global distribution of an application. Figure 1 shows the SGVis tool, with the ‘Apps’ option selected from the menu on the left. An analyst first selects an application from the drop-down list at the top, which is automatically populated with all the unique apps that have written to the SGLog database. Several tabs are available, the first being the overall number of users who have used the application each day since release. Separate trends are shown for the number of users who played for the first time that day (the green series), returning users who had used the application on a previous day (blue), and the total of these numbers combined (yellow).

![SGVis interface](image)

*Figure 1. SGVis shows several summary statistics on app usage. Here, the number of users running a trial application per day is plotted, with separate series for new and returning users.*

The graph shown in the figure is typical of the general trend we have observed for application releases. The peak of usage usually occurs on the first days of release. We have questioned participants on how they typically discover new applications, both via telephone interviews and through in-app questionnaires, and many have reported that they regularly browse the app repositories’ ‘Newest’ sections, where recently released applications are listed. This would help explain the early peak, with the number of new users gradually declining as an application falls further down the Newest list. The number of users per day often reaches a plateau—after around two weeks in Figure 1. A rise in the trend can be expected again when new versions of the application are released. Different repositories have varying policies on this, with some putting a release of a new version of an application at the top of the Newest list again. It is clear from the graph that there is always a steady stream of new users using the application, as represented by the green line. After the initial peak has plateaued, there is an average of around 100 new users every day. As the number of returning users (blue line) shows no significant increase, it can be inferred that there is a high degree of churn in the user base, with new users trying the application every day but the majority not returning regularly for several consecutive days or weeks.
Another section in the ‘Apps’ mode displays a map that presents an aggregated view of all the locations at which the selected application has been used. Maps are viewed via a live connection to Google Maps, with SGVis-supplied markers. In the left image in Figure 2, one unique marker is shown for every user who has used the application in the British Isles, placed at the location at which that user was most recently seen. Data can be filtered to show, for example, only those users exhibiting a specified minimum amount of usage. After a data set reaches a certain size, it becomes prudent to cluster the data before rendering it. This step aids in both keeping the Google Maps rendering time acceptable and creating a readable map, where it is possible to see user numbers. Figure 2 (right) shows an example of this, plotting users of Hungry Yoshi in Europe. Zooming into a map region expands a cluster to see the data represented as individual markers again. These maps use the open source MarkerClusterer utility (MarkerClusterer, 2011).

Viewing the geographic spread of an application’s users in this way provides an insight into the countries in which an application is popular. Researchers might like to act on this information in a number of ways. For example, iOS allows localisation of user interfaces. Developers can supply translations of all strings used in the user interface of an application, and these will replace the default English values if a user has selected that language on their device. If an application is seen to be receiving a lot of usage in a particular country, a researcher might consider it worthwhile to translate the application into that country’s native language in order to maximise usage. Indeed, it might be considered to be a researcher’s responsibility to the participants to translate terms and conditions into the languages of countries where the application is receiving a lot of use, in order that the users can make informed consent about taking part in a trial.

Figure 2. Map showing the geographical distribution of users of Hungry Yoshi, a mobile game released through an app store. Location data is clustered for large numbers of users to decrease rendering time and to aid readability.

Demographic information is available in another tabbed section. This includes device type and operating system version, which is automatically collected for every user, and age and gender of any participants who have chosen to submit that information via an in-app questionnaire. Figure 3 shows some of this information for Zoo Escape - a simple puzzle game application. The chart on the left shows the total
number of each type of iOS device on which the application was used. The total number of different
days on which a participant used the game has also been counted, and the chart on the right shows the
average number of days for each device type. From these two charts it can be seen that the application
has been used by more iPod Touch users than iPhone users, but that on average the iPhone users
returned to the game more often. iPad users play the game for the fewest number of days on average.

An application’s users can also be sorted and displayed in SGVis. The ‘top’ 50 users can be displayed,
using such measures as those making the greatest number of launches or accumulating the greatest total
minutes’ use. These values can be measured since release date or queried by a specific time period.

Figure 3. A breakdown of the iOS device types on which the Zoo Escape game was played (left) and the
average number of days users of each device type returned to the game (right).

Such a view of the user base is often useful in selecting users to contact for interviews, or in selecting
users to whom researchers can push specific questions via an in-app questionnaire. We showed in
(McMillan et al., 2010) that qualitative data can still be captured from users in a mass participation trial,
but that challenges exist in selecting the particular participants to interact with in this manner from the
large numbers available. A particular challenge is in contacting users at appropriate times in their
trajectory of usage of an application. In a more traditional study, it might be common to do interviews
shortly after a trial’s conclusion, while the application will still be fresh in the mind of participants. In a
mass participation deployment, however, there is not a clear definition of what ‘after’ a trial means.
Depending on whether the application has a server-side component, which may have maintenance costs,
an application might continue to work indefinitely. In that sense, a trial doesn’t have a fixed end point.
And during this time players will continuously start and stop using the application, so the overall highest
points scorers in a game might have ceased playing or uninstalled the application months or even years
earlier. In order to select those participants for whom use is still fresh in the mind, SGVis can be used to
see who is still using the application, perhaps by looking at the most active users in the last week, and
selecting potential interviewees from this subset.

Lists such as this, and maps showing markers for each user, can also be used to link to the second main
mode in SGvis, which allows for analysis of individual users in detail. An analyst might begin by
looking at an aggregate summary of an application’s data, then select an interesting-looking user to
smoothly drill-down to see visualisation of this single user in detail, as described in the following
section.
**Trial data viewed by user**

Where SGVis is more powerful than systems such as Flurry is in allowing detailed analysis of individual users. SGVis includes a number of tools for displaying information for a single user’s activity, which can be useful in identifying those that are unusual or interesting cases, or to prepare for further qualitative analysis, such as interviews.

Figure 4 shows an illustration of SGVis operating in this mode. An analyst can select a user to drill-down into from aggregate views, as described in the previous section, or simply enter a unique device ID (UDID). The analyst is then presented with information this user has declared, such as age and gender, and a table summarising all the SGLogged applications that the user under scrutiny has run. This table contains information such as last launch time and the number of days on which the user has run the application.

Clicking a row in this table displays a visualisation in the lower part of the screen that shows details of the selected participant’s usage of the selected application. The chart in the centre of the figure shows the times of day at which the user was running the application and the length of time it was running each day. Different days are shown along the x-axis, and the hours of the day are shown on the y-axis. A blue vertical bar is drawn to represent each period of usage. Underneath this, another graph shows the minutes of use per day, using the same x-axis as the chart above. The data in the figure is from a user of the Hungry Yoshi mobile game. It is notable that this user is a keen player of the game, using the application for almost 7 hours one day.

Whereas the tools described in the previous section aided evaluators in selecting appropriate users for interview, we suggest that these user-specific tools can aid evaluators in preparing for the interview. As an example of this, in studying usage of Hungry Yoshi, we were interested in the ways in which users fit playing the game into their everyday routines. Through use of these tools we noticed that one player had been using the application in short bursts during the evenings. On telephoning her to enquire about this behaviour, she revealed that she worked as a poker dealer in a casino, and used the game as a way of filling in time during her enforced breaks.

Another usage of these tools is in seeing when a participant has stopped using the application. On one occasion we noticed that one of our most keen users of a game had abruptly stopped playing. We attempted to telephone him to enquire why and although he did not have time to speak to us, he emailed us to say “I was in a rush on my break at work, where btw I'm banned for 30 days from bringing my iPhone into the building becase I was caught playing Hungry Yoshi.”
Figure 4. Individual users can also be studied in detail. The table at the top shows all the logged apps this user has run. The lower charts show the times of the day the selected app was run (the 24 hours of the day on the y-axis, shaded blue when in use) and below, the length of time the app was running each day.

User data can also be viewed by location. Whereas the maps shown in the previous section showed aggregated usage of an application and rendered one marker for each user, Figure 5 shows an example of mapping a single user’s data, where every location at which that user has run the application is displayed. If the user has run the app while moving or travelling, the map will show the route. This has allowed us to identify for example users who play the game while commuting, compared to those who only play at home and we have been able to target specific questionnaires towards these participants to learn more on this observed behaviour.
In general, the tools for analysing a single user’s activity are also useful for contextualising interesting behaviour. For example, when analysing our World Cup Predictor application, we were particularly interested in usage of the ‘head-to-head’ mode, which transferred data over ad hoc peer-to-peer Bluetooth connections. Occurrences of this activity were quite rare, only being performed by 45 of the 10,806 registered users. On each occasion the head-to-head feature was used, an analyst could study the context in terms of location and time of day, and also see if this contrasted with the player’s general pattern of usage. For example, it could be seen whether the player had travelled to a friend’s house to perform a Bluetooth transfer.

Similarly, our applications all have a section for reporting bugs to developers. SGVis reports the user’s device type and operating system version and it can be invaluable in diagnosing a bug to view a user’s context when the problem was discovered, as well as a history of usage.

**Live trial data**

The previous sections described analysis of data accumulated over the duration of a trial. SGVis can also visualise ‘live’ data, which is a regularly updating view summarising recent activity in SGLogging applications. Figure 6 has an example of this. A slider on the bottom of the application allows the analyst to set the period of time to view, ranging from activity recorded in the last 1 minute up to the last 24 hours.

Two graphs in the top left of the screen show respectively the number of users to have used each application in the specified time period and, below, which users have been active, charting the number of logged actions. A map in the top right shows the locations where people have been playing recently. Again, this is zoomable to see areas of interest in more detail. Finally, a table at the bottom of the screen shows data in a more raw format as it comes in. Data is uploaded to the database continually by people using the apps worldwide, and SGVis refreshes its view every 10 seconds.
Figure 6. SGVis has a constantly updated view of recent activity

All the sections in the live mode are linked, to afford brushing and linking between views (Becker & Cleveland, 1987). This allows users to make selections in one view which will then be reflected in the others. For example, an analyst can select an app by clicking on a bar in the bar chart, and this will filter all other views to show only the data recorded from within that app. Similarly, the analyst can draw a box to select a particular area of the map and this will filter the other views to show only activity recorded in the selected region.

By using the SGLog and SGVis toolkit, analysts are no longer required to wait until the conclusion of a trial before retrieving log data, which means they can react to events they see in the log data and possibly change aspects of the trial as it is ongoing. This could include updating the software based on observed usage, or perhaps altering the logging code to fine-tune the specific data being captured.

There are also uses in monitoring a trial to ensure that everything is running smoothly. For example, when a new version of an application has been released, it can be important to ensure that everything is working as intended for all users. As an example, using the live tool tool we once observed that the Hungry Yoshi application was showing an uncharacteristically small amount of activity. On further investigation, by moving back to look at data from the whole user base, it appeared that users with one
particular combination of model of iPhone and operating system version were unable to use the application following a recent update. Once the problem had been correctly diagnosed and the bug removed, a further update was released and we could use the live tool again to observe usage return to normal.

Although such monitoring of a trial is useful, it is probably impractical for a research project to have someone dedicated to looking at this data all the time. Future work will look at ways to automatically detect changes in patterns of activity and push out alerts to analysts.

**Derived data and user categorisation**

The final menu option in SGVis allows for more in-depth statistical analysis of data gathered from mass participation studies. A previous section described tools for studying individual users in detail. However, when there might be tens or hundreds of thousands of people using an application, with hundreds joining every day, it becomes impractical to study every user in detail. Tools included in SGVis aid analysts in this regard by clustering participants into different groups based on characteristics of usage. Thereafter, different groups of users could be handled differently, for example sent questionnaires with only those questions pertinent to their specific use of the application.

To illustrate this functionality, a use case is described showing results from analysis of Hungry Yoshi. The SGLog database is processed to create a number of statistical measurements for each participant. These include, for example, the cumulative time spent playing the game, time between first and last uses, type and form of response to feedback question, cumulative distance travelled when actively playing the game, and the size of the geographical area in which users play the game.

This data is used to create a high-dimensional vector for each user. An analyst can plot each of these dimensions $x$ vs $y$ to see trends or correlations, but practical experience has suggested it might be more useful to consider the full high dimensional vector as a whole, and look for ways of being able to analyse the vectors all together.

One way to achieve this is through the use of ‘spring model’ dimensional reduction algorithms (Eades, 1984; Chalmers, 1996; Morrison, Ross & Chalmers, 2003) to map the high dimensional vectors to a 2D layout, which we can then search for patterns and relationships across all aspects of the data. Such algorithms take as input the set of high-dimensional vectors representing the trial users, and create a 2D layout such that the high-dimensional pairwise relationships between vectors are well represented in 2D space. This is achieved through simulating a system of mechanical springs connecting objects, with each spring’s ideal rest distance proportional to the high-dimensional dissimilarity between the pair of users. By iterating a process of calculating and applying the forces exerted upon the objects by each spring, objects are pulled and pushed towards a state of equilibrium that should be a good representation of the relationships between each pair of users.
Figure 7. A spring model layout of data derived from the Hungry Yoshi game. Views are linked, so that selecting an area of the layout on the right highlights the corresponding rows of the table on the left.

Figure 7 shows an example of this in SGVis. The full data is shown in a table and a scatterplot is on the right. Here an analyst can choose to plot any pair of dimensions as x against y or, as shown in the figure, perform a dimensional reduction routine to derive a 2D layout. Again, the views are linked, so that making a selection in one highlights the corresponding subset of objects in the other.

To perform this type of analysis in greater detail, data can be exported for analysis in the Hybrid Information Visualisation Environment (HIVE) (Ross & Chalmers, 2004), a Java-based multi-purpose visualisation tool that has been extended to handle SGVis data files. Figure 8 shows an example of a HIVE analysis, where various clustering and layout techniques have been used to create a basic categorisation of users. 16 logged features from each user are processed in a spring model to generate a 2D layout. The analyst has then used Voronoi-based clustering (Okabe, Boots, Sugihara & Chiu, 2000; Ross, Morrison & Chalmers, 2004) to group coherent subsets of this layout, resulting in four distinct categories, as shown in the scatterplots in the top right of the figure.
Figure 8. Each dot in this spring model layout (top middle) represents 16 logged features from one of 13,000 users of the Hungry Yoshi game. A Voronoi diagram was created from this layout, and then an interactive clustering algorithm (based on thresholding Voronoi cell sizes and spacing) was used to create four major categories of styles of play.

Having assigned each user to one category, the high-dimensional centroids of each of the created clusters can be computed, which show the analyst the trends shown by the ‘average’ user in each group. Cluster centroids were studied by creating a parallel coordinate plot (Inselberg & Dimsdale, 1990), where the dimensions under scrutiny are stacked as vertical axes and each of the four clusters is represented by a polyline intersecting these axes at the appropriate value. This is shown in Figure 9.

In this example, one group is clearly distinct from the others, rating highly in terms of physical activity as well as temporal activity. It would appear that these are the game’s most enthusiastic users, who spend the most time playing, and who also play over the widest area and travel the furthest while playing. This group also has the high response rate to questionnaires. Another group (characterised as ‘commuters’) are distinguishable from the others by having a higher mean speed, due to greater amounts of travel while playing. A third group (‘static players’) has spent the smallest proportion of time moving while playing, and have the smallest play area, and a fourth group (‘beginners’) comprises those who have played the game for a briefer period and who have cumulated less distance playing and offered fewer responses to questionnaires.
Figure 9. The centroids of the four clusters identified in Figure 8 have been calculated, and displayed here in a parallel coordinates plot. Studying the characteristics of the four groups leads us to label these categories ‘top players’, ‘commuters’, ‘static players’ and ‘beginners’.

By creating such categories, it is hoped that steps can be taken to help cope with an unmanageable volume of users. In a more traditional trial with a smaller user base, a researcher would possibly want to study each user in detail. This becomes less practical when user numbers are in the tens of thousands, and an analysis tool that can summarise the major different ways in which participants have used the trial system could greatly reduce this burden. Of course, researchers may then wish to study individual participants from each of the identified categories, to see usage in greater detail.

Categories can also be used as a basis for further evaluation, with, for example, different questionnaires being shown to each of the four groups. Once categories have been identified and questionnaires written, this could be an automated process, with a process on the server analysing users’ behaviour and distributing the appropriate questionnaire at preset times. In the future our aim is to go further still, experimenting with deploying different software updates to users based on their categorisation. Other ongoing work is on coupling the SGVis toolkit more tightly to HIVE.

**DISCUSSION**

As ubicomp trials scale up in size, with a vast global user base replacing the more traditional local deployment of software, analysts can be challenged by the large amount of data generated by tens or hundreds of thousands of users. Online tools such as Flurry can generate statistics about app usage, but fail to provide details on individuals’ use that may be important to researchers.

The SGVis analysis tool, alongside the SGLog framework which can be used to quickly instrument software running on mobile devices, allows analysts to easily generate aggregate statistics and visualisations, viewing data by application or by user. Such tools can aid in conducting quantitative analysis, taking advantage of the large numbers for use in statistical calculations. Qualitative analysis need not necessarily suffer either in trials of this nature. We described in (McMillan et al., 2010) how researchers running a large scale trial still performed interviews arranged in social networking applications and conducted via VoIP services, with the sort of data visualisations provided by SGVis valuable in allowing analysts to select interviewees and brief themselves on a participant’s usage of the application before calling him or her. A ‘live’ mode allows for near real-time monitoring of application use across the globe. Ongoing work in this area is looking to augment applications with a messaging
feature, so that analysts could push a query out to a user in real-time, which would pop-up on a device running a trial application. In this way, researchers could ask users about a specific event they had just observed, while the context of the event was still fresh in the user’s mind.

It is hoped that tools of these forms, used together, can support the software development lifecycle. We suggest a scenario where, in monitoring live data, researchers witness an interesting event. Logging code is altered to capture this event in more detail, and the updated logging code deployed to participants. Collected data is processed through dimensional reduction and clustering techniques to identify a category of users behaving in the observed manner, some of whom are contacted for interview. New software updates are created to support the observed behaviour and deployed to the members of the appropriate categories. Researchers continue to observe usage of the new software. In our future work, we aim to look in more detail at visualisation tools for developers, that might use abstractions such as populations (Chalmers, 2010) to understand complex data consisting not only of patterns of use but also patterns of application software structure.

Another area of future work would be to trial SGVis by studying its use among other research groups. This would not be a major technical difficulty; by instrumenting SGVis itself with the SGLog software, we would be able to collect data on the use of the analysis software in the same way we do with the mobile software. By collecting this type of data we could then conceivably use SGVis in the analysis of itself. Although we have found it very easy to get large amounts of users for mobile applications, researchers running large-scale trials of iOS-based research software are obviously a far smaller target population. Therefore we encourage anyone interested in using SGLog and SGVis to help in the analysis of iOS-based research software to contact us.

In summary, it is hoped that through the use of tools such as SGVis, researchers will be able to manage the vast amounts of data that can be generated by app store-style deployment. In particular, we aim to combine both quantitative and qualitative forms of evaluation, to support timely response to ongoing trial activity as well as retrospective analysis of temporal patterns, and directly assist new software development as well as evaluation work. In such ways, we aim to obtain the full benefits offered by mass-scale research.

REFERENCES


