Context Data | ECA Rule Engine | Log Platform

Technology Reports

User-created Application Sharing Platform Enabling Users to Create Their Own Activity Support Applications

Because a variety of sensor and log data are available on smartphones, there are environments in place that offer context aware applications that adapt to the circumstances of the user. However, with these applications, there is a substantial burden placed on the user associated with making settings to match usage scenes, and there have been difficulties with reusing existing applications. To solve this issue, we have prototyped a user-created application sharing platform that enables the user to modify application behavior just by making settings on a template, and to share the application with others easily. This prototype enables users who have no programming knowledge to create applications, and use them for activity support for themselves, friends and acquaintances.

Research Laboratories
Tomohiro Nakagawa
Chiaki Doi
Kazuhisa Sekine
Ken Ohta
Hiroshi Inamura

1. Introduction

Due to the inclusion of a variety of sensors and open-source operating systems in smartphones in recent years, it has become easier to create context aware[*1] applications that adapt to user circumstances. Context aware applications use acceleration sensors and operational log data etc. to adapt to the context - the time, the user's location, whether the user is moving or still and so forth - which enables the application to offer information or services that the user wants with appropriate timing [1].

Because context aware applications can provide appropriate information and services even if there is no explicit user operation, they can be applied for activity support for the user themselves or other users. Hereinafter, a context aware application that is designed for user activity support is referred to as an "activity support application."

For activity support applications to recognize a context, or execute functions in response to a recognized context, the necessary data must be configured in advance. These settings are a burden for users who are not familiar with using smartphones. Furthermore, because there are a variety of contexts that could be recognized, there are issues with maintaining flexibility and ease-of-use in describing and specifying the contexts that should be recognized to personalize an activity support application to suit the user.

In this article, we characterize a usage model that includes the application configurator and "Event-Condition-Action (ECA)" rules [2] to support multi-perspective semantics[*2] of "who," "when," "where" and "what/how," and propose a common user-created appli-

*1 **Context aware**: Context is data that indicates time, the user's location and circumstances. "Context aware" means that context data is used to automatically execute processing in response to user circumstances.

cation platform to solve these issues.

2. Service Image and Related Technologies

2.1 Activity Support Applications

As an example of an activity support application, **Figure 1** shows a service image of a guide application. In this example, the event organizer distributes the guide application to event participants in advance. By recognizing the context of having arrived at the event on time from the participants' position data, the guide application notifies the event organizer of the event participants' arrival. At the same time, the application presents a locality map and related information to the event participants. On the other hand, if it seems that a participant is going to be late to the event opening, the application displays a dialogue with the contact details of the event organizer, prompting the event participant to contact the organizer. In this way, activities such as confirming user participation or attendance are supported without the user having to perform any explicit operations in the application.

In this article, "creation" of activity support application is defined as composition of application by combining the information and functions necessary for activity support. By specifying contexts that should be recognized and functions that should be executed in response to contexts, activity support methods can be customized for individual application users. The data that is required for an activity support application is called "settings data." In the guide application example described in Fig. 1, the event opening date and time, venue and organizer contact details etc. are used as settings data.

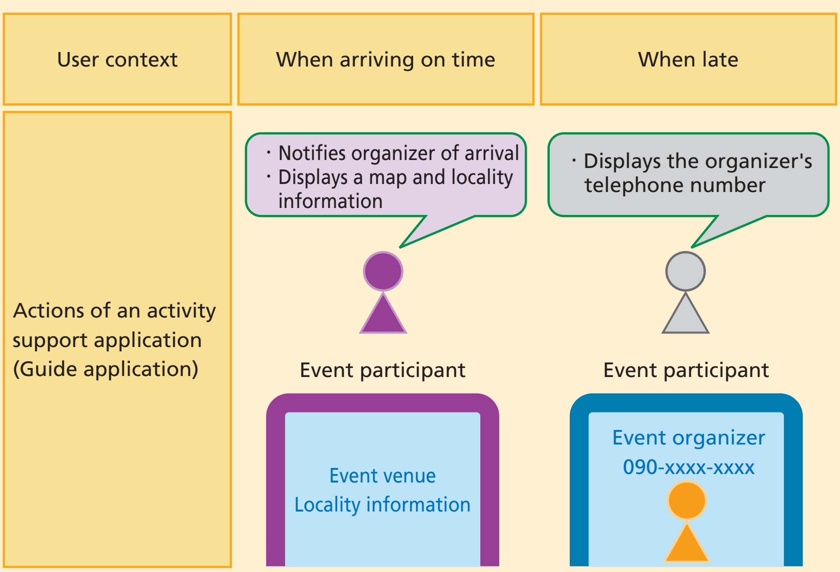

Figure 1 Activity support application image

2.2 Related Technologies

There are several technologies based on Android™[*3] platform for creating applications by assembling functions such as MIT's App Inventor[*4] [3], Microsoft's on{X}[*5] [4], and BLOCCO from NTT DOCOMO and GClue[*6] [5].

With App Inventor, applications are created by combining function blocks in an editor. In on{X}, applications are created in a web browser by setting process triggers such as time, location, weather conditions and so forth, and setting the processes to be executed by those triggers. With BLOCCO, users can create service mash-ups by specifying terminal conditions such as screen ON or remaining battery power, and services and functions to be executed in response to those conditions. All these technologies are aimed at creating applications by assembling functions.

In addition to enabling users to create applications, the main thrust of this research is to enable activity support for other users who are not familiar with using smartphones. We also aim to enable customization by enabling contexts to be recognized from a variety of perspectives, and are deepening investigation into creating activity support applications derived by reusing existing ones.

---

*2 **Semantics**: Terminology that indicates the meaning of data.

*3 **Android™**: A software platform for smartphones and tablets consisting of an operating system, middleware and major applications. A trademark or registered trademark of Google Inc., United States.

*4 **App Inventor**: An application development environment that enables Android applications to be created, by combining components visually on a browser. Published by the Massachusetts Institute of Technology.

3. Requirements

3.1 Function Requirements

1) Reducing Configuration Burden

Before commencing use of an activity support application, it is necessary to input settings data that the application references to recognize contexts and execute functions in response.

Problems arise with inputting settings data if the user does not have the required information or is not motivated to perform the settings. For example, as guide application users, event participants might not know the event organizer's contact details.

Therefore, for event participants to be able to easily use an activity support application, there is a need to eliminate complications at the time of use.

2) Maintaining Flexibility with Context Recognition

In providing activity support in diverse contexts, it is necessary to recognize contexts which are expressed by a range of semantics using sensor and system log data. Contexts can be recognized by analyzing the various system log and sensor data that terminals are able to acquire. For this reason, there is a need to include analytical functions designed for different types of data and different data characteristics.

3.2 Performance Requirements

To reduce terminal power consumption, it is also necessary to constrain resources consumed with context recognition. For this reason, we aimed to limit processing time for user context recognition to 300 ms or below for activity support applications created with this system. Although there is a dependency on the frequency of context recognition, this 300ms standard enables quick return to sleep mode after executing context recognition processing, which we believe will sufficiently minimize the amount of power used.

4. User-created Application Sharing Platform

4.1 Applications Based on ECA Rules

We adopted script-based ECA rules for activity support applications created with this system, rather than Java®*7 applications. ECA rules are rules for processing methods consisting of combinations of events, conditions and actions, and are described as XML (eXtensible Markup Language)*8 files in this system.

To execute ECA rules, the activity support application user must first install the ECA rule engine in his or her terminal. **Figure 2** describes terminal architecture. Downloaded activity support applications are loaded on the ECA rule engine. As well as the ECA rule engine, the terminal also has a log function and a log DB, and collects the log and sensor data required to execute the activity support application.

4.2 Application Creation and Usage Model with Application Configurator

To meet the requirement of reducing configuration burden, we have

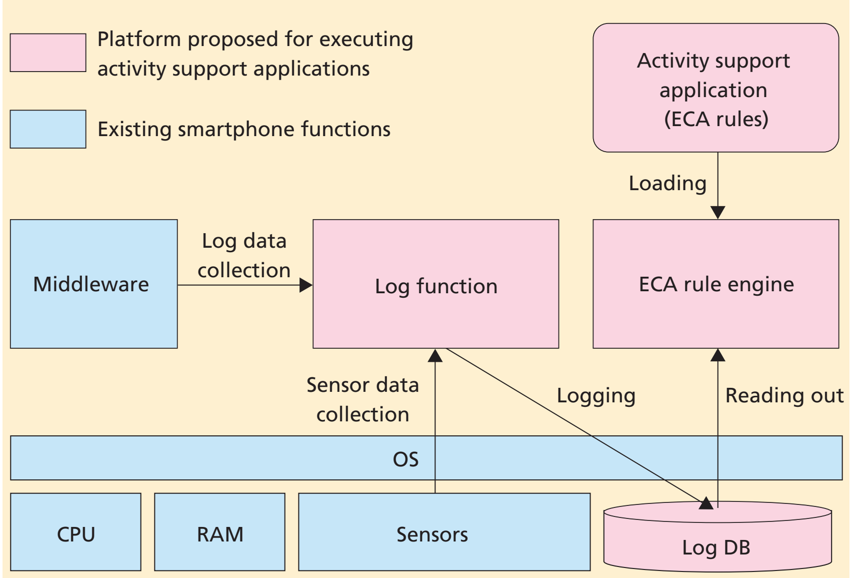

Figure 2 Terminal architecture

*5 **on{X}**: A browser-based application development environment that enables Android applications to be created by using recipes prepared as application templates. Published by Microsoft through Google Play™. Google Play™ is a trademark or registered trademark of Google, Inc.

*6 **BLOCCO**: BLOCCO is a trademark of GClue, Inc.

*7 **Java®**: An object-oriented programming language. Applications implemented in Java are executed on a virtual machine so they can run on different platforms. Oracle and Java are registered trademarks of Oracle Corporation, its subsidiaries and affiliates in the United States and other countries.

decided upon a system that generates the ECA rules that comprise the activity support application just by configuring a template. Templates are lists of typical functions that can be incorporated into activity support applications.

There are three parties involved in the creation of activity support application - the template developer, the application configurator, and the application user. In the guide application example, the application configurator is the event organizer, and the person who uses the application is the event participant. **Figure 3** describes the application creation and usage model, including the application configurator. Compared to conventional usage models, our usage model is characterized by the positioning of the application configurator between the template developer and the application user (Fig. 3), in order to reduce configuration burden of the application user.

Figure 4 shows an image of creating and using the activity support application. In this application creation and usage model, the application configurator performs template settings on behalf of the application user. The configurator selects context recognition functions, and sets the data required to execute the selected functions. In the guide application example, the event organizer, as the application configurator, sets information such as the event start time and venue for the event participant. Thus, the event participant does not need any knowledge about activity sup-

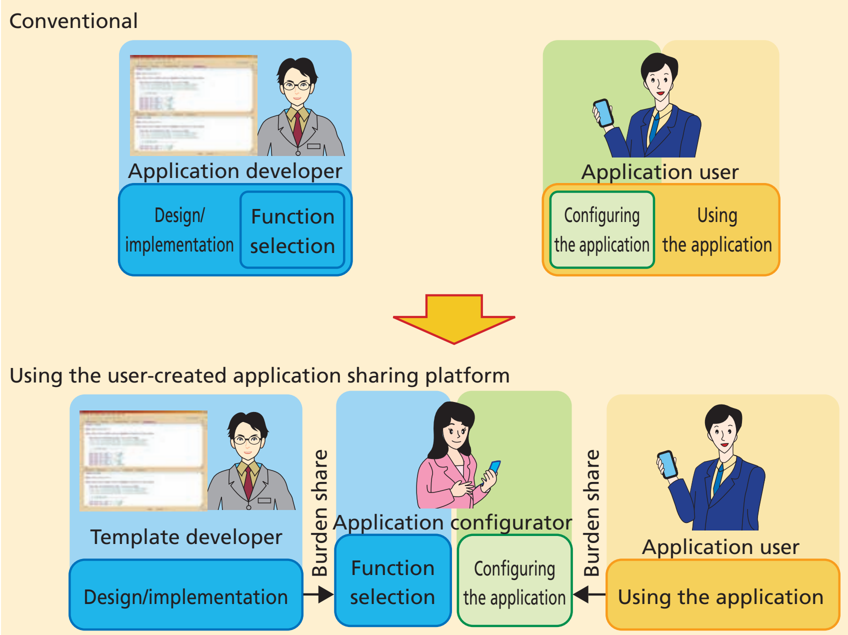

Figure 3 Application creation and usage model including the application configurator

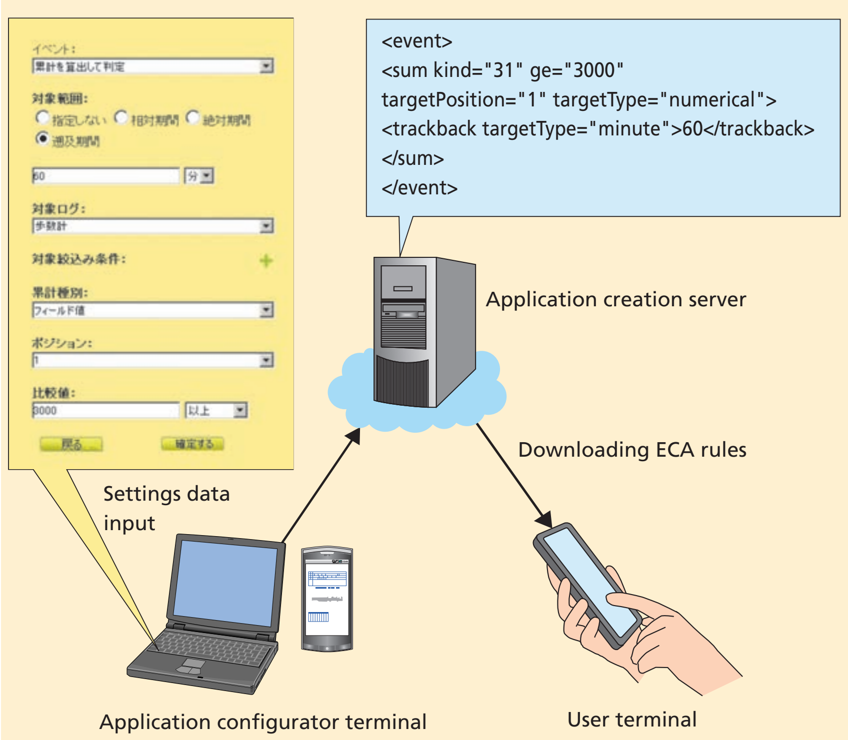

Figure 4 Creating and using activity support applications

*8 **XML**: A markup language proposed by the World Wide Web Consortium (W3C) for describing the meaning and structure of documents and data. It can be extended by user-defined tags (see *9).

port related to the event, because the event organizer, who is motivated to make the event-related settings, performs the settings instead of the event participant having to do it. In this way, the user can use the activity support application without entering any information at all.

4.3 ECA Tag Specifications to Provide Flexibility

To meet the requirement for flexibility with context recognition, we defined ECA tag[*9] specifications (**Table 1**) to enable multi-perspective context recognition of "who," "when," "where" and "what/how." These perspectives have already been suggested in existing research as "W4H" (who, when, where, what and how) - all of which should be covered by a context aware service [6]. For example in the guide application, position data describing "where" can be described using the CENTER tag, as shown below. The system detects when the user is within a radius specified in meters of the longitude and latitude of the location specified in the CENTER tag.

<center lat="35.224702" lon="139.671356" kind="21">
<le type="numerical">100</le>
</center>

In the same way, detection rules for "who," "when" and "what/how" can be set using other tags [7].

The ECA tag specifications that we propose need to support diverse log types. To this end, the specifications enable matching of numerical and character strings, and analysis of terminal log generation frequency and generation interval. This enables the application configurator to customize settings data and logs to be recognized, and create an activity support application that is personalized for the application user.

4.4 ECA Rule Processing Performance

To assess processing performance, we evaluated rule processing time with the SUM & SUB tags — tags that require particularly longer processing time with context recognition than other tags. The specifications of the terminal we used for evaluation are listed in **Table 2**.

The SUM & SUB tags provide analytical functions for log data collected in the smartphone. The SUM tag adds up the number of times log events have occurred within a specified time range, and compares the value to a threshold. The SUB tag measures the time interval between pairs of different log types, for

Table 1 ECA tag specifications for events and conditions

Class	Tag type	Parameters	Description
Who	OCCUR	Terminal log type Field value	Trigger as a log event containing a specified person's ID
When	TIME	Date and time	Trigger as a specified time
Where	CENTER	longitude, latitude, radius	Trigger as entry into the area around a specified location
What, How	OCCUR	Terminal log type Field value	Trigger as a log event in the terminal that has a specified field value.
	RANGE	Terminal log type Field value range	Condition as log events within a specified range
	MATCH	Terminal log type Field value character string	Condition as log events with the specified character string
	SUM	Terminal log type, Period, frequency	Condition where the number of log events reaches the specified number
	SUB	Terminal log type, Period	Determines whether conditions are met for the interval between two types of log event.

Table 2 Specifications of terminal used for evaluation

CPU clock	RAM	Flash ROM	Android version
1.0GHz	512MB	1,024MB	2.3.4

*9 **Tag**: A descriptive method of indicating headings and links on Web pages.

example, the screen ON as start logs and screen OFF as finish logs. In this case, the total time that the screen is on can be determined. Because these tags require analysis of logs accumulated in specified time span, they have a larger impact on processing time than other tags.

Figure 5 shows the results of SUM tag evaluation by changing the target time span from 15 minutes to two hours. The results measured for the SUM tag processing time were less than 100 ms. Furthermore, the results for the SUB tag evaluation exhibited similar rule processing times, thus, we were able to confirm that processing time is 100 ms or less. In the SUM tag evaluation results, the average processing time for a single log entry in the accumulated log was 2.0 ms seconds, with a maximum of approximately 4.7 ms. From these results, processing of the SUM and SUB tags is possible at or below the required 300ms even when 150-entry logs are accumulated in normal status, and is also satisfied with 62.7 entries if processing speed has been retarded.

From the above results, because the ECA tag processing time meets the performance requirements, we believe that power consumption can be sufficiently reduced.

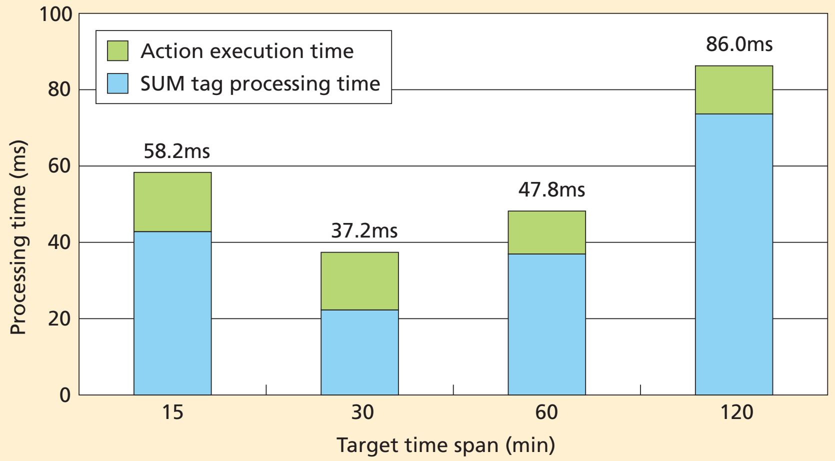

Figure 5 Activity support application processing time

5. Conclusion

In view of the arguments presented in chapter 4, we have proposed a platform with specifications that meet the set requirements.

In this article, we have proposed a user-created application sharing platform for creating activity support applications only by setting templates, and for sharing the application with other users.

This system will enable people with no programming knowledge to support the activities of others through activity support applications.

Into the future, we will further develop energy-saving measures for context recognition processing in the rule engine such as detecting the area where the user is located. Also we will introduce privacy protection mechanisms for handling context data.

References

[1] L. Daniele, P. D. Costa and L. F. Pires: "Towards a Rule-Based Approach for Context-Aware Applications," EUNICE'07, Jul. 2007.

[2] P. D. Costa, J. P. Almeida, L. F. Pires and M. Sinderen: "Evaluation of a Rule-Based Approach for Context-Aware Services," IEEE GLOBECOM, 2008.

[3] MIT App Inventor. http://www.appinventor.mit.edu/

[4] Microsoft: "on{X}." https://www.onx.ms/

[5] BLOCCO. http://www.blocco.jp/

[6] R. F. B. Neto and M. G. C. Pimentel: "Toward a Domain-Independent Semantic Model for Context-Aware Computing," Third Latin American Web Congress, 2005.

[7] T. Nakagawa, C. Doi, K. Ohta and H. Inamura: "Customizable Context Detection for ECA rule-based Context-aware Applications," ICMU, May 2012.