Dealing with Uncertainty in Context-Aware Mobile Applications

Christian Jung, Andreas Eitel, Denis Feth, Manuel Rudolph
Fraunhofer IESE
Kaiserslautern, Germany
Email: {christian.jung, andreas.eitel, denis.feth, manuel.rudolph}@iese.fraunhofer.de

Abstract—**The exploitation of context-awareness, especially in mobile devices bears a huge potential. For example, mobile workers benefit from systems that adapt security settings to the current situation. However, context-aware computing strongly relies on raw data from various sources that might be neither trustworthy nor authoritative. In this work, we present a context model that explicitly reflects security and relevance of context information sources in order to improve context detection. We introduce a security rating denoting the trustworthiness of the context information, i.e., its vulnerability to forgery, and a relevance rating denoting the source's decisive impact on context detection.**

Keywords–Context-Awareness; Context-Modeling; Security

I. Introduction

Context-awareness of software applications is still in its infancy although it has been researched since the beginning of the nineties. Recently, the rise of mobile technologies introduced a new class of devices with various sensors providing context information. For such devices, context-awareness can be particularly useful to adapt user interfaces or security measures to the current situation. For example, context-awareness enables more flexible control by limiting the applicability of security measures only to situations where they are indispensable.

An important building block for enabling context-aware security is context modeling. The user's contexts (i.e., his current activity and situation) have to be determined by aggregating (low-level) contextual information, such as current location, battery consumption, or connectivity of his mobile device. In order to provide reliable decision support, context descriptions have to be trustworthy and accurate.

To leverage the full potential of context-awareness for IT security, it is essential to identify security-relevant contexts and to reliably detect these contexts. Thus, the context evaluation must furthermore consider information about how easy it is to counterfeit contextual information.

To this end, a methodology for eliciting and modeling contextual information is needed that yields reusable and comparable context descriptions. In particular, this method must support the identification of suitable context information sources and the aggregation of low-level pieces of context information into an overall context description.

In this work, we present our context model and context descriptions that explicitly include a relevance and a security rating for each context information source. This rating enables us to provide quality statements for the accuracy of context detections. Security decisions benefit from the quality statements within a context description, aggregated during runtime.

The paper is structured as follows: Section II addresses related work in the area of context definition and context information modeling. Our context modeling approach is presented in Section III, followed by Section IV addressing uncertainty of context information sources. In Section V, we apply our approach to an example scenario. Finally, Section VI provides a summary and an outlook on future work.

II. Related Work

This section provides an overview of the state of the art in context-awareness and context-aware computing. More specifically, the term *context* and its early definitions are introduced and different context modeling approaches are described.

A. Definitions of the term "Context"

The notion of *context* emerged over time, the earliest definitions in our sense originate in the early nineties. All of them share similarities, but they also show differences. We will present the most prevalent and influential definitions.

In 1994, an early definition was provided by Schilit et al. [1], stating that a context is characterized by three aspects: where you are, whom you are with, and what resources are nearby. Therefore, Schilit et al. infer that context-aware systems have to depend on the location of use, nearby people, hosts, and accessible devices, as well as on changes over time.

In 1997, Brown and Bovey [2] describe context similar to Schilit et al. but include temporal attributes, such as time of day, season, and temperature, as additional contextual information sources. In addition, the authors propose to enrich context by using additional (user-provided) information to obtain more valuable information for their application.

Hull et al. [3] describe context as "many aspects of a user's situation", such as "user identity, location, companions, vital signs, air quality, and network availability". Franklin and Flachsbart [4] focus on intelligent environments observing their users. They state that context-aware computing should consider the observed situation of the user. A similar description can be found in [5]. Ryan et al. [6] state that context should include location as well as states of external and internal sensors of the computer itself. Hence, they also consider virtual context sources such as the state of the software running on the device. Pascoe [7] also considers virtual context sources, but describes them as the states of the application and its environment rather than states of the computer itself. Pascoe et al. [8] reveal the more rich and complex nature of context and that context can be complex. Furthermore, in accordance with other publications, they state that context is more than just location. Hofer et al. [9] partition context information regarding its origin and differentiate between physical, virtual, and logical context information. Physical context information, such as location, acceleration or light intensity, can directly be measured by sensors. Such physical measures are described as low-level context sources that are continuously updated. Virtual information stems from user data or internal system data. The latter context category, logical context information, is obtained by combining physical and virtual context sources according to some abstract logical rules.

Our work does not introduce a new context definition; rather, we use a combination of existing definitions and descriptions of context. Similar to [9], we partition context sources into virtual and physical information sources, depending on the origin of the information, and we logically link them. As this work focuses on mobile devices and their users, activities of the user are an important aspect, as well as the operational state of the device. Thus, we define context as:

> *Context is the state of all context information sources (including virtual and physical sources) that characterize the activity of the user and the operational state of the mobile device in a specific situation at a certain time.*

B. Modeling Context Information

Context-aware systems strongly rely on the quality of the context information, which is usually represented in a context model. The modeling and provision of context information is very important to fulfill the desired task. In this work, context awareness aims at the enforcement and adaptation of flexible security policies on the mobile device and its applications. Different approaches for modeling context information have been suggested. In [10] and [11], the authors survey the most relevant approaches and classify them into five categories:

Key-Value Models are the simplest model for structuring context information. As such models provide no structuring of information, they are easy to manage. They are often used, although they provide only limited support for more sophisticated modeling [10][11]. Key-Value models allow easy querying by simple algorithms matching the key value pairs. The querying can be enriched by Boolean operators or wildcards for the matching algorithm.

Markup Scheme Models use a hierarchical structure of markup tags containing attributes and their values. A well-known example is the eXtensible Markup Language (XML). A markup scheme has been proposed, for instance, in [12]. In contrast to key-value models, markup scheme models provide a mechanism for structuring context information.

Graphical Models can strongly vary in their representation. The best-known representative is probably the Unified Modeling Language (UML), which is also suitable for modeling context, as shown in [13] or [14]. Such models are easy to understand for human beings, but often lack formality (except for UML). Henricksen et al. [15] present a context extension for the Object-Role Modeling (ORM) approach. An interesting aspect of their model is the differentiation between static context information (i.e., facts that remain unchanged as long as the entities they describe persist) and dynamic context information. They distinguish between contextual information that can be treated as property or constant attribute and changing contextual information such as location.

Object Oriented Models provide their information as a collection of objects that contain context information. Such models can employ all object-oriented modeling techniques such as encapsulation, reuse, or inheritance. The objects can represent different context types and provide interfaces for the retrieval and processing of their context information. Hofer et al. used such object oriented models for the Hydrogen context framework [9].

Logic Based Models use formal methods to specify context information and rules that can be applied to them. Hence, they usually provide a high degree of formality. Typically, in the reasoning process new facts can be derived based on known facts and a given set of deduction rules. Albeit being very formal and precise, profound logic-based modeling is quite hard and modeling given facts can become very complex. One such approach has been published in 1994 by McCarthy and Buvac [16].

Ontology Based Models are used to represent concepts and interrelations. They are a very promising instrument for context modeling, especially with the option to apply ontology reasoning techniques and automatic derivation of new relationships. A representative of this class of models is the Aspect-Scale-Context (ASC) model, which is based on the Context Ontology Language proposed by Strang et al. [17].

For further details, the reader is referred to two surveys: Baldauf et al. [11] survey existing context systems and frameworks, including their respective context models. Another survey by Bettini et al. [18] describes the state of the art in context modeling and reasoning. In summary, the decision which kind of modeling approach to choose can only be made by investigating the underlying application scenario and the context to be modeled. Regardless of the presented approaches, none of them consider the uncertainty in context detection and are thus unsuited for security purposes. In our work, we use a combination of the presented context modeling approaches (hybrid approaches) and extend them with two quality attributes to deal with uncertainty.

III. CONTEXT MODEL

Our context model specifies relations between the context, the user, and the mobile device. Figure 1 presents a macroscopic view of the core parts of our model and their interrelations.

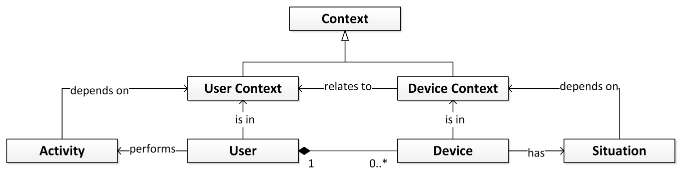

Figure 1. Context Model (Macroscopic Viewpoint).

A user can perform an activity at a certain time, and the user context depends on the performed activity. However, an activity change does not have to imply a context change. If we assume a user context such as *Traveling*, for instance, the user can perform several activities that belong to this context: *driving a car*, *riding a train*, *taking the plane*, or *walking*.

Hence, the relation between activity and user context depends on the granularity of the modeled user contexts and activities. Moreover, the technical abilities of the context information retrieval limit the detection options. In an ideal world, the user's activities and user contexts would always be congruent. However, the boundaries between certain activities are fluid and often cannot be determined by context information sources. Accordingly, the differentiation between activity and user context will persist. Hence, the context model has to cope with such uncertainties.

The situation of the device includes internal states of the device and attributes of the environment the device is in. Similar to the relation between activity and user context, device context and situation would always match in an ideal world. However, due to technical limitations, several distinct device situations lead to the same device context. It is apparent that the environment and internal states of the device can only be sensed by context information sources that are technically available. Thus, device context and user context must be detectable by existing context information sources, but are only approximations of the real world, neglecting unmeasurable information. In contrast, activity and situation strive to represent the world as it really is.

The core part of the context model is the context itself. The goal is to model the user and device context as an abstraction and aggregation of pieces of information obtained from various context information sources.

A. Context Description Structure

To model contexts, we use *Expressions* as generic statements that can be combined to form arbitrarily complex descriptions (see Figure 2). Expressions can be combined and nested in a way that the respective overall result is a Boolean value with additional ratings for security and relevance (see Section IV). We have a *GenericExpression* component that forms the basis for all other types of Expressions (arithmetic, comparison, and logic) and a *ConstantExpression* component holding a constant value. The Expression interface has a method for evaluating itself and a method for retrieving the return type. For type safety, it is important to have these type assignments, as a context description, at least in theory, could combine any expression type. However, there is a check whether the relation is allowed. For instance, a comparison between a Boolean type and a list of values would be rejected.

A specialization of the GenericExpression is the BinaryExpression, which allows exactly two subordinated expressions.

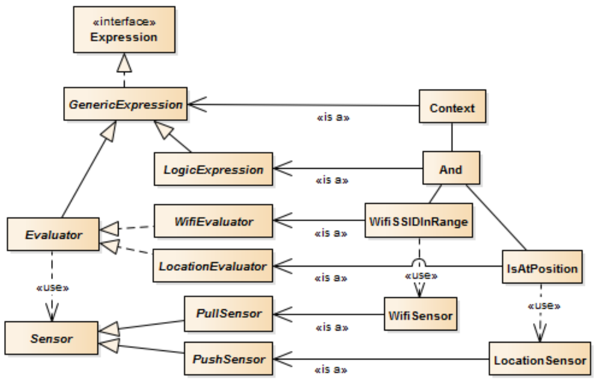

Figure 2. Excerpt Expression Model.

ComparisonExpressions, for instance, take exactly two sub-expressions for their evaluation.

ArithmeticExpressions are expressions for addition, subtraction, multiplication and division. For evaluating the expression, all assigned sub-expressions are joined by the appropriate operation. In general, ArithmeticExpressions can contain an unlimited number of nested expressions.

Similar to ArithmeticExpression, a *LogicExpression* can take an unlimited number of nested expressions for evaluation. For the moment, *and*, *or*, and *not* with their usual semantics have been implemented. Future extensions could include fuzzy logic or other evaluation capabilities.

The *EvaluatorExpression* can express various behaviors, for example, by using comparisons or arithmetic operations. The remaining task is to obtain concrete data from the system under observation. This is done by *Evaluators*.

Evaluators provide an abstract representation for context information sources, which usually deliver low-level information, such as the current coordinates of the Global Positioning System (GPS) or acceleration values. Each evaluator can be configured by several parameters that influence its behavior. Evaluators can be included as sub-expressions at arbitrary locations within the presented structure.

An evaluator encapsulates one or more context information sources. In general, we distinguish between push and pull behavior, based on the characteristics of the underlying sensors the Evaluator encapsulates. Some sensors can be configured to push new information as soon as new data is available (e.g., acceleration or location sensors). Other sensor information has to be pulled to obtain current information (e.g., mobile device settings, calendar). Hence, the configuration depends on the sensor type. Pull sensors contain a *scheduleInterval* parameter specifying the frequency of data updates. For example, wireless network information can be requested every five minutes.Push sensors usually contain parameters to configure conditions when data updates will be delivered. For example, the location sensor will only deliver new information if the mobile device location changed by at least a specified distance.

Our preferred format for a context description is XML (see Figure 3). The context id attribute is mandatory and has to be unique, as it is used to reference the context. In general, a configuration can contain several expressions of types arithmetic, comparison, or logic (future extensions could extend the list of available expression types).

The specification in Figure 3 includes two different evaluators: a GenericLocation evaluator for checking a specific location and a WiFiIsSSIDInRange-Evaluator to scan for specific wireless Service Set Identifiers (SSID). The location evaluator consumes the following parameters: location values (i.e., *latitude* and *longitude*), *distance* (specifying the data delivery and distance accuracy), *provider* (location information from network and/or GPS) and *maxAge* (allowed data age for evaluation). The wireless evaluator uses the parameter *ssid* to scan for this specific wireless SSID. The listing also shows an example of arithmetic expressions and comparisons. Evaluators may contain a relevance and a security rating, which are described next.

```
<context id="example-context">
  <logic:and>
    <logic:or>
      <evaluator name="GenericLocation" relevance="5">
        <param name="latitude" value="49.431479"/>
        <param name="longitude" value="7.7520288"/>
        <param name="distance" value="15.0"/>
        <param name="provider" value="0"/>
        <param name="maxAge" value="3600"/>
        <param name="resultType" value="boolean"/>
       </evaluator>
       <evaluator name="WiFiIsSSIDInRange" relevance="3">
         <param name="maxAge" value="60"/>
         <param name="keepEnabled" value="true"/>
         <param name="scheduleInterval" value="15"/>
         <param name="ssid" value="wlan-home"/>
         <param name="resultType" value="boolean"/>
        </evaluator>
    </logic:or>
    <!-- Arithmetic demo: 2*2*4 >= 10+(36/6) -->
    <comparison:greaterEqual>
      <arithmetic:multiply>
        <constant type="double" value="2"/>
        <constant type="double" value="2"/>
        <constant type="double" value="4"/>
      </arithmetic:multiply>
      <arithmetic:add>
        <constant type="double" value="10"/>
        <arithmetic:divide>
          <constant type="double" value="36"/>
          <constant type="double" value="6"/>
        </arithmetic:divide>
      </arithmetic:add>
    </comparison:greaterEqual>
   </logic:and>
</context>
```

Figure 3. Evaluator Example.

IV. Dealing with Uncertainty

We introduce two quality metrics to address uncertainty of contextual information: a *security rating*, denoting the difficulty for an adversary to counterfeit the measurement of the context information source and a *relevance rating*, expressing the value of the context information source for the identification of the overall context.

A. Security Rating

Every evaluator has a security rating assigned to it. The rating takes the values from one (very low) to five (very high). Basically, the security rating denotes the trustworthiness of the context information, i.e., its vulnerability to forgery. The security rating within our work is defined as follows:

> *The Security Rating is a global indicator expressing difficulty and challenge for an adversary to counterfeit a context information source.*

The following guidelines are used to rate an evaluator. A security expert or group of experts have to perform this task when an evaluator is developed. The experts have to assess the necessary preconditions for a successful counterfeit context information source:

(i) insider knowledge or configuration details
(ii) special expertise or knowledge to perform the operation
(iii) special software or application
(iv) special hardware or equipment
(v) influence on information source (backend or environment change)

Based on these prerequisites, the metric to determine the rating for every evaluator is defined as follows:

- **1 (very low)**: 0 out of 5 prerequisites needed
  It is easy to counterfeit the measured values (e.g., just change or enter the value). An example for such a rating is the time of the mobile device or calendar entries of the user.
- **2 (low)**: 1 out of 5, but not prerequisite (iv)
  The manipulation of the sensed value can be done with little effort. An example is to simulate a high light intensity with a torch or to shake the device to forge acceleration values.
- **3 (medium)**: 2 out of 5 OR 1 out of {(iii), (iv), (v)}
  Some preparations are required, but they are not really challenging. An example would be faking the SSID or BSSID of a WiFi hotspot, which can directly be done by using a second smart mobile device.
- **4 (high)**: 3 out of 5 OR 2 out of {(iii), (iv), (v)}
  The required measures are challenging for an adversary, and without special knowledge, it would not be possible to perform the attack. An example is to simulate that the device is connected to an encrypted (mobile) network.
- **5 (very high)**: 4 out of 5 OR 3 out of {(iii), (iv), (v)}
  Forging of context information sources requires deep knowledge about the internal configuration and significant expertise; moreover special equipment is needed, such as software and hardware. An example is the GPS sensor or cell phone tower information, for which an attacker would need special hardware and knowledge how to use it.

The security rating has to be defined once, and it has a global scope for every instantiated context tree. However, it is possible to manually change the rating by explicitly setting it in the evaluator tag of the context description. For example, the security rating of a virtual context information source can vary according to its trustworthiness. A read-only enterprise calendar will be much harder to counterfeit than the personal calendar maintained by the user itself. Hence, we can manually assign a higher security rating to the evaluator using the read-only enterprise calendar.

B. Relevance Rating

Every evaluator has a second rating assigned to it, expressing the contribution of the context information source to the overall context identification. Similar to the security rating, it accepts values from one (very low) to five (very high). The rating represents whether the provided information tends to be decisive or has a less authoritative impact on context detection. The relevance rating is defined as follows:

> *The Relevance Rating is a local indicator expressing the correlation of a context information source with an activity, or situation.*

This rating depends on the modeled context, but also on the quality of a sensor and cannot be specified by just following generic guidelines. The retrieval of the relevance rating is part of the automatic derivation of context descriptions [19]. Essentially, we use different statistical methods to correlate sensor data with activities and use the results (e.g., a correlation matrix) to determine the relevance of a sensor for the characterization of an activity or situation. Furthermore, we use the calculated results to define the structure of our context descriptions.

C. Context Evaluation

The evaluation result of a context is obtained by evaluating the tree structure of the context description. The logical operators have their standard meaning.

- **OR**-relation: $A \vee B$ is true if A is true, or if B is true, or if both A and B are true.
- **AND**-relation: $A \wedge B$ is true if A is true and B is true.
- **NOT**-relation: $\neg A$ is true if A is false.

The calculation of the relevance and security rating for a context is as follows:

- **AND/OR**-relation: All fulfilled quality attributes of the elements in an AND/OR group affect the overall relevance or security rating. They are summed up to the denominator. The quality attributes of all evaluators that are actually fulfilled in the system under evaluation are summed up to the numerator.
- **NOT**-relation: The quality attribute of the element is propagated to the parent node, if the subordinated expression is false.

The quality attributes ensure that the fulfillment of those evaluators with highest relevance or security rating has the strongest impact on the overall result. The security policy specification bears responsibility for defining suitable thresholds for the security and relevance ratings that are sufficient to trigger a change of the security settings. Furthermore, the decision strongly depends on whether to tighten or to ease security restrictions.

V. Application Scenario

Our approach has been applied in a company which administrates mobile devices via the mobile device management (MDM) solution *MobileIron*. Via MobileIron, they adjust security settings (e.g., to impose password restrictions or storage encryption, to install or revoke certificates for virtual private networks, or to disable camera or microphone), and perform actions such as sending messages to the user or wiping the device. However, these settings and actions are rather static and cannot be adapted according to the current operational state of the device or the user activity. In this setting, context-awareness can provide more flexibility. For example, camera and microphone usage can be prevented within company premises, but allowed elsewhere. However, when used for security purposes, context detection has to be accurate and reliable in order to comply with company regulations.

MobileIron provides different types of policies and a label mechanism to assign policies to mobile devices. *Security policies* control security behavior such as password restrictions of the mobile device; *lockdown policies* limit the use of the mobile device such as disabling Bluetooth, camera, or microphone. We specified two security policies ($security1$ and $security2$) and three lockdown policies ($lockdown1$, $lockdown2$, and $lockdown3$), which are shortly described in the following.

TABLE I. Security Policies

	security1	**security2**
Maximum Inactivity Timeout:	30 min	2 times
Maximum Number of Failed Attempts:	3 min	5 times

TABLE II. Lockdown Policies

	lockdown1	**lockdown2**	**lockdown3**
Bluetooth:	Disable	Enable (Audio only)	Enable
Camera:	Disable	Enable	Enable
Microphone:	Disable	Enable	Enable
NFC:	Disable	Enable	Enable
Screen Capture:	Disable	Enable	Enable
Lockscreen Widgets:	Disable	Enable	Enable
USB Debug:	Enable	Disable	Enable

In Table I, $security1$ has higher priority than $security2$. Hence, if both policies are activated, security1 would be used. In Table II, the lockdown policies have the following priorities: lockdown1 $>$ lockdown2 $>$ lockdown3.

Despite the policies, we specified multiple labels: $default\_label$, $work1\_label$ (tighten security), $work2\_label$ (ease security) and $home\_label$. The assignment of policies to labels is depicted in Figure 4.

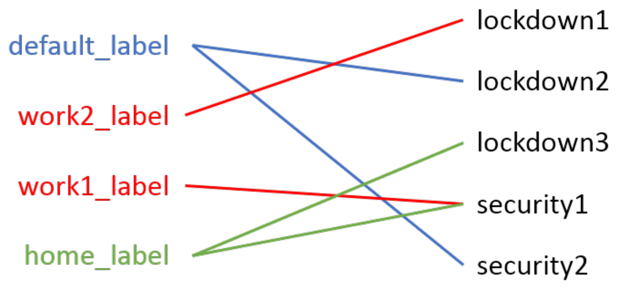

Figure 4. Mapping between Labels and Policies.

In the described setting, two contexts are of relevance: “Home” and “Work”. Both contexts are modeled by using wireless, location, calendar, and time evaluators. Figure 5 illustrates the context tree for the “Work” context c_1, including all assignments for the security and relevance ratings. As the security rating has a global scope for each evaluator type, the value does not change within the same type of evaluator. In contrast, the relevance rating changes, depending on the results from the statistical calculation.

For example, the wireless network *wlan-staff* has the highest statistical significance (correlation result), followed by *wlan-guest* and *wlan-extern*. This is reflected in the final relevance ratings of the wireless evaluators. *wlan-staff* is the employer’s wireless network and has the highest relevance. The calendar seems to be a relevant context factor, but as users usually do not schedule their entire working day in the calendar, the calendar evaluator has the lowest relevance. Similar behavior holds for the time evaluator. The company has flexible working hours, but the core working hours are between 09:00am to 4:00pm. Hence, users arrive earlier or stay longer at work, to reach their daily working time. Nevertheless, the time evaluator is more relevant than the calendar evaluator.

The company has defined several policies assigned to the

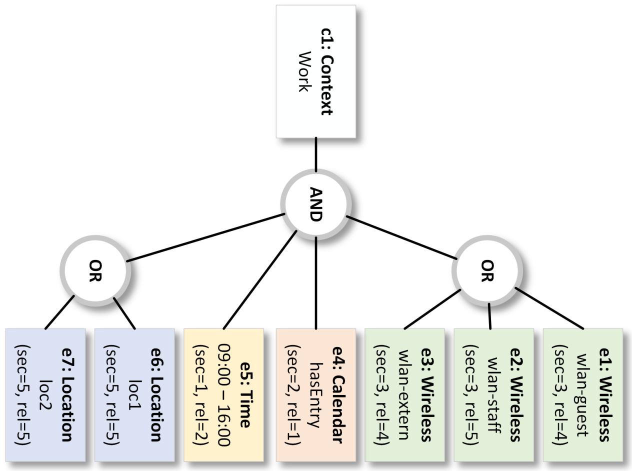

Figure 5. Context Tree Example for Context "Work".

"Work" context, on which we focus in the following. On the one hand, there are policies easing security restrictions of the mobile device at work (in favor of usability). For example, the company increases the display timeout at work to thirty minutes for usability reasons ($work1\_label \rightarrow security1$). On the other hand, there are policies tightening the security at work. For example, the company prohibits the usage of camera, microphone, etc. at work to meet organizational policies ($work2\_label \rightarrow lockdown1$). The idea is now to define appropriate thresholds to reflect company needs.

To tackle this, we calculated the following cases:

- Context c_1 is $true$ with highest relevance $\rightarrow 1.00$
- Context c_1 is $true$ with lowest relevance e_1, e_4, e_5, e_6 are $true \rightarrow 0.46$
- Context c_1 is $false$ with highest relevance $e_1, e_2, e_3, e_5, e_6, e_7$ are $true \rightarrow 0.96$
- Context c_1 is $false$ with lowest relevance $\rightarrow 0.00$

Hence, the relevance range for c_1 $is\ true$ is between 0.46 and 1.00, and for c_1 $is\ false$ is between 0.00 and 0.96. Analogously, we calculated the security rating for c_1. Figure 6 shows our policy state chart and the state change criteria to activate and deactivate the policies. To change the states by using the relevance and security rating, as well as the fulfillment of the context, allows us to model hysteresis behavior. For example, changing from the state $work1\_label$ (inactive) to $work1\_label$ (active) is harder than changing from the state $work1\_label$ (active) to $work1\_label$ (inactive).

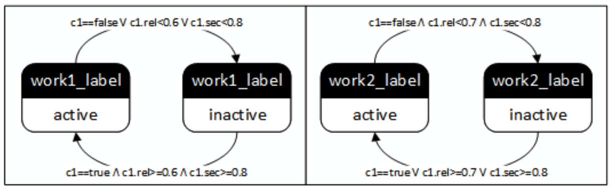
c1==false V c1.rel<0.6 V c1.sec<0.8
work1_label
active
work1_label
inactive
c1==true Λ c1.rel>=0.6 Λ c1.sec>=0.8
c1==false Λ c1.rel<0.7 Λ c1.sec<0.8
work2_label
active
work2_label
inactive
c1==true V c1.rel>=0.7 V c1.sec>=0.8

Figure 6. Policy Statechart.

For instance, assume that $work1\_label$ is inactive, the evaluators e_1, e_4, e_5, e_6 are true, and the evaluators e_2, e_3, e_7 are false. This results in context c_1 to be true. However, the relevance is only 0.46 and the security is only 0.50, which would prevent the change from inactive to active. Hence, we need at least one more evaluator changing its state to true for reaching the relevance threshold and even one more for reaching the security threshold. Vice versa, let us suppose that $work1\_label$ is active and the evaluators have the same state as before. Although c_1 is still fulfilled, we would make the change as the ratings are below the defined thresholds of 0.60 (relevance) and 0.80 (security).

A. *Lessons Learned*

The practical application was performed by two of our internal researchers, as they had to manually observe the mobile devices. For our case study, we used a Samsung Google Nexus 10 running Android 5.1 and a Samsung Galaxy Tab (SM-P600) running Android 4.4.2. Both devices were added to the MDM solution. We made several observations in our practical application, which we describe next.

Some sensors have uncertainties and inaccuracies in their measurements. In our scenario these are the location and the wireless sensors. We configured the wireless sensor to scan for specific wireless networks every five minutes. However, the sensor occasionally misses some wireless networks although the networks are available. If we measure for a specific wireless network ten times, the sensor will miss this specific network one to two times. Hence, we have a failure rate of 10 to 20 percent in our measurements. Let's assume our context "work" c_1 is fulfilled and $work1\_label$ is active as well as $work2\_label$ is active. The affected wireless evaluators are e_1, e_2, e_3. All other evaluators are assumed to be fulfilled, i.e., working correctly. As they are in OR-relation, all three have to provide a wrong measurement to yield an overall evaluation result of c_1 that is wrong, which did not happen during our evaluation period. Evaluator e_2 has the highest impact on the relevance and security ratings. If the evaluation of e_2 fails, the relevance rating is at 0.81 and the security rating is at 0.86. Both ratings are above the specified thresholds to trigger a state change for the labels $work1\_label$ and $work2\_label$. The failure of an additional wireless evaluator, for instance e_1 or e_3, puts the ratings to 0.65 (relevance) and 0.73 (security), which are below the thresholds. Such ratings result in a change of $work1\_label$ from active to inactive, which is uncritical as it tightens our security settings. Regarding $work2\_label$, we will stay in the active state, as the overall context c_1 is still true, which is also uncritical. To trigger a state change, we would need all three evaluators to fail, which did not happen during our evaluation period, as already mentioned.

Regarding the location evaluation, we observed that we have some uncertainties in the location evaluation of e_6 and e_7 when people are entering the specified locations. Such location changes usually happened in the morning, when people arrived at work, and after noon, when people came back from lunch outside the company. The reason for detection failures is the inaccurate location fix after a location change. The Android location services return a coarse grained location, which is outside our specified locations for the evaluators. Let's assume our context "work" c_1 is not fulfilled and $work1\_label$ is inactive as well as $work2\_label$ is inactive. The affected wireless evaluators are e_6, e_7. All other evaluators are assumed to be fulfilled, i.e., to work correctly. To trigger a state change, both evaluators have to be evaluated to true. Regarding

$work1\_label$, it is uncritical as we are remaining in the high security settings; however, $work2\_label$ is critical. As we configured to receive a location fix latest every five minutes and after location changes greater than fifteen meters, we may stay five to ten minutes in a wrong state. However, as the Android location services pushes new information after the location fix, we observed to stay less than five minutes in the wrong state.

We modeled all evaluator groups in AND-relation, which was a bad decision regarding the time and calendar evaluators. As the working hours was given between 09:00am and 04:00pm, e_5 was also configured to be true in the specified interval. However, people usually do not completely stick to these working hours. We observed that $work1\_label$ stayed too long in state inactive (starting working before 09:00am) or changed from active to inactive too early (working longer than 04:00pm). Similar observations were made for the calendar evaluator e_4. We learned to model evaluators lower relevance rather in OR-relation than in AND-relation. However, we have to gain more experience to make a final decision.

VI. Conclusion and Future Work

We presented a model for representing security-relevant contexts as context descriptions. The model contains a security rating for each evaluator to quantify the overall trustworthiness of the context description during runtime. In addition, the model provides a relevance rating expressing the conduciveness of a context information source to the overall context. The context descriptions enable context-aware security decisions by referencing them as a decision criterion in security policies.

Future work will investigate potentials to return additional data types as an overall context result. Enriched context types facilitate the use of contextual information in the decision making process and improve expressiveness for security policy specifications. Presently, context descriptions are specified manually before being activated and are therefore rather static. Future work will investigate how context descriptions can be parametrized at runtime. This may include the use of context results as parameters for other context descriptions.

Regarding the security and relevance rating, we will further extend our evaluation criteria. We realized that faking the presence of contextual information can be easier in some cases then faking its absence (e.g., it is easier to simulate the SSID of a wireless access point than jamming the beacons from an existing one). Finally, we will explore the inclusion of accuracy information into our model as an additional quality attribute for judging the reliability of context information.

Acknowledgment

This paper is a result of the Software Campus project KoSiUX (Kontextsensitivität zur Steigerung der Sicherheit und User-Experience mobiler Anwendungen), which is funded by the German Ministry of Education and Research, grant number 01IS12053, and the European project SECCRIT, funded by the European Commission in the context of the Research Framework Program Seven (FP7), grant agreement number 312758. The authors are responsible for their content.

References

[1] B. Schilit, N. Adams, and R. Want, "Context-aware computing applications," in Proceedings of the Workshop on Mobile Computing Systems and Applications. IEEE Computer Society, 1994, pp. 85–90.

[2] P. Brown and J. Bovey, "Context-aware applications: from the laboratory to the marketplace," IEEE Personal Communications, vol. 4, no. 5, 1997, pp. 58–64.

[3] R. Hull, P. Neaves, and J. Bedford-Roberts, "Towards situated computing," in Proceedings of the 1st IEEE International Symposium on Wearable Computers, ser. ISWC '97. Washington, DC, USA: IEEE Computer Society, 1997, pp. 146–153.

[4] D. Franklin and J. Flachsbart, "All gadget and no representation makes jack a dull environment," In Proceedings of AAAI 1998 Spring Symposium on Intelligent Environments. AAAI TR SS-98-02, 1998, pp. 1–6.

[5] A. Ward, A. Jones, and A. Hopper, "A new location technique for the active office," Personal Communications, IEEE, vol. 4, no. 5, 1997, pp. 42–47.

[6] N. S. Ryan, J. Pascoe, and D. R. Morse, "Enhanced reality fieldwork: the context-aware archaeological assistant," in Computer Applications in Archaeology 1997, ser. British Archaeological Reports, V. Gaffney, M. van Leusen, and S. Exxon, Eds. Oxford: Tempus Reparatum, Oct. 1998.

[7] J. Pascoe, "Adding generic contextual capabilities to wearable computers," in Proceedings of the 2nd IEEE International Symposium on Wearable Computers, ser. ISWC '98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 92–99.

[8] J. Pascoe, N. Ryan, and D. Morse, "Issues in developing context-aware computing," Handheld and ubiquitous computing, 1999, pp. 208–221.

[9] T. Hofer, W. Schwinger, M. Pichler, G. Leonhartsberger, J. Altmann, and W. Retschitzegger, "Context-awareness on mobile devices - the hydrogen approach," in Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS'03) - Track 9 - Volume 9, ser. HICSS '03. Washington, DC, USA: IEEE Computer Society, 2003, p. 292.1.

[10] T. Strang and C. Linnhoff-Popien, "A Context Modeling Survey," in Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England, 2004, pp. 1–8.

[11] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems." International Journal of Ad Hoc and Ubiquitous Computing, vol. 2, no. 4, 2007, pp. 263–277.

[12] N. Ryan, "ConteXtML: Exchanging Contextual Information between a Mobile Client and the FieldNote Server." in Computing Laboratory, University of Kent at Canterbury, CT2 7NF, UK, 1999, pp. 1–8.

[13] Q. Z. Sheng and B. Benatallah, "Contextuml: A uml-based modeling language for model-driven development of context-aware web services development," in Proceedings of the International Conference on Mobile Business, ser. ICMB '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 206–212.

[14] J. Bauer, "Identification and modeling of contexts for different information scenarios in air traffic," 2003.

[15] K. Henricksen, J. Indulska, and A. Rakotonirainy, "Generating context management infrastructure from high-level context models," in In 4th International Conference on Mobile Data Management (MDM) - Industrial Track, 2003, pp. 1–6.

[16] J. McCarthy and S. Buvac, "Formalizing context (expanded notes)," Computer Science Stanford University, Stanford, CA, USA, Tech. Rep., 1994.

[17] T. Strang, C. Linnhoff-Popien, and K. Frank, "Cool: A context ontology language to enable contextual interoperability," in LNCS 2893: Proceedings of 4th IFIP WG 6.1 International Conference on Distributed Applications and Interoperable Systems (DAIS2003). Volume 2893 of Lecture Notes in Computer Science (LNCS)., Paris/France. Springer Verlag, 2003, pp. 236–247.

[18] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni, "A survey of context modelling and reasoning techniques," Pervasive and Mobile Computing, vol. 6, no. 2, 2010, pp. 161–180.

[19] C. Jung, D. Feth, and Y. Elrakaiby, "Automatic derivation of context descriptions," in 2015 IEEE International Inter-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2015.