A Context-Oriented Framework for Software Testing in Pervasive Environment

Heng Lu
The University of Hong Kong
Pokfulam, Hong Kong
hlu@cs.hku.hk

Abstract

In this article, we present our test framework for assuring pervasive software applications that overcomes the identified challenges faced by conventional testing techniques. The framework will serve as a basis for automating the test process, as well as to assist testers in generating adequacy test sets or finding test oracles.

1 Motivation

Pervasive computing has attracted a lot of studies and investigations in recent years driven by the wide usage of smart devices, sensor networks, and RFID (Radio Frequency Identification) technology. Context-awareness is one of the most significant features of pervasive computing, in which surroundings of computing entities are modelled by extensive usage of *contexts* that abstract the environmental attributes, and the behavior of applications will adapt to the changing contexts. While there has been significant progress in design models and implementations of pervasive computing nowadays [1, 4, 8], however, we observe the inadequate work in testing and analyzing pervasive software applications. Our work is to explore new challenges in testing these pervasive programs, and to seek for solutions to tackle these challenges. In particular, we aim at building up a framework that either automates the testing process of pervasive applications, or provides testers with guidelines in generating test cases or judging test outcomes when a manual approach is inevitable under certain circumstances.

Many researchers and practitioners recognize RFID to be one of the enabling technologies for the next generation of software application. Still, RFID-enabled applications should tolerate environmental errors and deal with unreliable contexts captured from their environment because RF (radio frequency) signals, among others, often vary continually in the complex physical environment. For instance, the environmental effect may make a reader unable to sense a particular RFID tag, even though both of them are functioning properly [3]. It may mislead the RFID readers to read the tag accidentally, while the tag is out of range [3]. In the following, we identify features of such pervasive systems that raise software testing challenges.

Streaming contexts from environment: In pervasive computing, an application may sense contexts from the environments *in streams*. For instance, in response to RFID reader signals, an RFID tag will disseminate RF signals to the readers, and thus the application, nearby. Its signal emission rate is high; the related context variables will thus change often. For the same parametric inputs, such a program may alter its execution paths based on the received contexts, which makes the application adaptive to the environment. However, this makes it hard to replay the execution if test cases are input values only.

Scattered context-relevant operation: In pervasive computing, there exist at least two dimensions to update or use contexts. First, computational entities have egocentric behaviors. They often communicate with their environment through the contexts they are aware of [4]. Second, for a single computational entity, software developers may use the popular layer architecture to encapsulate context management from the usage of contexts into distinct layers to ease the related application development. Both dimensions pose challenges to the standard integration testing, in which static analysis of the method invocation hierarchies may be hard to calculate effectively.

Uncertainty of computation: We have explained in [2] that it is often hard to define the test oracles, i.e. expected outcomes of computation, for pervasive applications. Informally, the programs interact with the changing environment; and it is hard to control certain environmental quantities as planned. This makes it hard to specify the test outcomes precisely in advance.

2 Proposed Solution

Redesign test cases with streaming context: In order to replay the program behavior with a certain test case, we plan to redesign the format of test cases to capture the streaming environmental updates. In an ordinary test case, each variable is represented as a symbol denoting the program input or its initial value. However, for context-aware

29th International Conference on Software Engineering (ICSE'07 Companion)
0-7695-2892-9/07 $20.00 © 2007 IEEE

programs, we propose that each context variable in a test case is represented as a vector, with each element recording the context value at each reference occurrence. For instance, the test case $\langle x = 1, \vec{c} = \langle (s_0, 11), (s_3, 10), (s_9, 12) \rangle \rangle$ requires that the initial value of the ordinary variable x is 1, while for the context variable c, the initial value should be 11 and the referenced values at statements s_3 and s_9 should be 10 and 12, respectively. In testing, redesigned test cases that use vectors to capture streaming context data are expected to replay the same execution paths so that an adequacy test set can be reused with the same level of coverage.

Derive context data flow for structural testing: In a generic view of pervasive environment, entities communicate with external surroundings only via contexts. We propose to model how contexts are to be used and defined in programs at various software-implemented tiers, which may consist of filtering, repairing, reasoning, and using of contexts [8], so as to consolidate all context-relevant operations into the same picture. Based on the derived context data flow information, novel test adequacy criteria can be designed for the generation of test sets and coverage metrics [6].

Utilize necessary properties for assessing computation outcomes: In dealing with the test oracle problem, let us consider the RFID location sensing application [7] as an example. With the assumption that the environment has similar offset effects on the estimated results in adjacent areas [2], we can derive a necessary property as

> P1: *"For two points with actual distance of 1 meter, the distance of their estimated locations should be within the range* $[1-\mu, 1+\sigma]$*."*

where μ and σ denote predefined error tolerance levels. Any computation result that violates P1 is regarded as exposing an error. The property can serve as a guideline to generate a series of test cases. For instance, after selecting the first test point with coordinates $(1,1)$, a list of follow-up test points with coordinates such as $(1,2)$, $(1,0)$, and $(0,1)$ can be selected according to P1.

3 Evaluation

In our preliminary work in [6] and [2] we conduct experiments to evaluate our proposal of test adequacy criteria and necessary properties as test oracles, respectively, on the *Cabot* middleware system [8] with the *LAMDMARC* RFID location sensing application [7]. In [6] we report that our best criterion *all-pairwise-du-associations* has a 37% improvement with the benchmark *all-uses* criterion in detecting faults with reasonable failure rate (between 0.002 and 0.012). In [2] our proposed necessary property is able to detect 71.4% of the seeded faults whereas the original estimated locations can hardly be compared directly with the corresponding actual locations.

We will extend our approach in [6] with the redesigned test cases to incorporate the requirements of streaming contexts. We also plan to study other benchmarks, such as random approach or adaptive random approach [5], to deal with the open system nature of pervasive environment. Also, the tester's effort in designing various necessary properties will also be quantified and assessed for a more realistic proposal.

4 Conclusion and Expected Contributions

This article describes our proposal of a novel framework for testing software applications in the generic pervasive computing scenarios. We identify new testing challenges for pervasive applications and propose the following solutions: (a) a novel format of test cases to facilitate replay of context-aware programs and reuse of test cases, (b) adequacy test criteria based upon context data flow, and (c) selection of necessary properties serving as test oracles for imprecise and uncertain computation in pervasive environment. The framework, with the prototype tool to be evaluated, is expected to integrate the proposed solutions to either automate the testing process of assessing test coverage and test outcomes, or provide guidelines when manual approaches cannot be completely waived in generating adequacy test sets or test oracles.

References

[1] L. Capra, W. Emmerich, and C. Mascolo. CARISMA: context-aware reflective middleware system for mobile applications. *IEEE TSE*, 29(10):929–944, 2003.

[2] W. K. Chan, T. Y. Chen, H. Lu, T. H. Tse, and S. S. Yau. Integration testing of context-sensitive middleware-based applications: a metamorphic approach. *IJSEKE*, 16(5):677–703, 2006.

[3] S. R. Jeffery, M. Garofalakis, and M. J. Franklin. Adaptive cleaning for RFID data streams. In *Proceedings of VLDB 2006*, pages 163–174, 2006.

[4] C. Julien and G.-C. Roman. Egospaces: facilitating rapid development of context-aware mobile applications. *IEEE TSE*, 32(5):281–298, 2006.

[5] F. C. Kuo, T. Chen, H. Liu, and W. K. Chan. Enhancing adaptive random testing in high dimensional input domains. In *Proceedings of SAC 2007*, 2007.

[6] H. Lu, W. K. Chan, and T. H. Tse. Testing context-aware middleware-centric programs: a data flow approach and an RFID-based experimentation. In *Proceedings of SIGSOFT 2006/FSE-14*, pages 242–252, 2006.

[7] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil. Landmarc: indoor location sensing using active rfid. *ACM Wireless Networks*, 10(6):701–710, 2004.

[8] C. Xu, S. C. Cheung, and W. K. Chan. Incremental consistency checking for pervasive context. In *Proceedings of ICSE 2006*, pages 292–301, 2006.

0-7695-2892-9/07 $20.00 © 2007 IEEE