UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
Facoltà di Ingegneria
Corso di Studi in Ingegneria Informatica

tesi di laurea

Gui testing automatico di applicazioni Android tramite emulazione di input ed eventi provenienti da sensori

Anno Accademico 2011/2012

relatore
Ch.mo prof. Porfirio Tramontana

correlatore
Ing. Domenico Amalfitano

candidato
Nicola Amatucci
Matr. 885/337

Dispositivi mobili Android e sensori

- Dispositivi mobili Android come smartphone e tablet sono corredati di un insieme abbastanza vario di sensori fisici (accelerometro, bussola, …)
- Vengono utilizzati sia per interagire con l'utente, sia per reagire a variazioni nell'ambiente (ad es. regolazione della luminosità dello schermo)
- Lo stack software Android permette di semplificare la gestione dei sensori
- Variazione nel valore dei sensori possono influenzare la GUI
- Quando si effettua il testing GUI-based è opportuno dunque tenere in considerazione gli eventi sensore

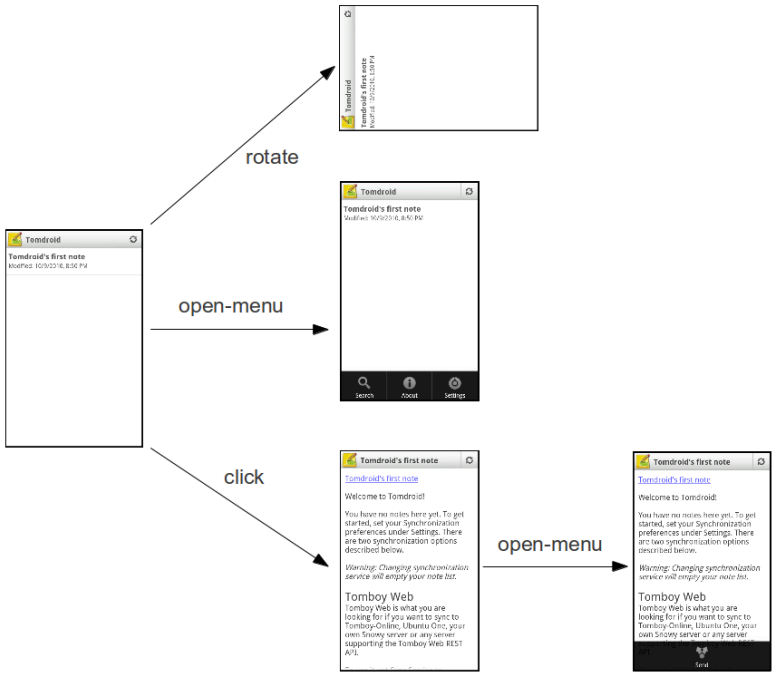
Android Testing Framework

- Android mette a disposizione un'estensione del framework jUnit per effettuare il testing delle applicazioni
- Si possono scrivere sia test jUnit classici, sia dedicati alla piattaforma Android
- Il collegamento con l'applicazione sotto test avviene tramite l'Instrumentation Framework
- L'Instrumentation Framework permette di:
  - Controllare l'applicazione
  - Controllare l'ambiente in cui l'applicazione viene eseguita

Android Testing

- Strumenti inclusi nel SDK
  - **Monkey**: Random-testing della GUI e Stress-testing
  - **Emma**: Copertura del codice
- Strumenti standard de-facto
  - **Robotium**: helper per l'interazione con la GUI
  - **Roboelectric**: testing sulla JVM locale di applicazioni Android
- **Android Ripper**

Android Ripper



- è uno strumento di automazione del testing GUI-based
- è configurabile e implementa sia tecniche crawler-based che di random testing
- è basato su un approccio black-box
- permette di effettuare crash testing, testing di regressione, smoke testing e di valutare la copertura dei casi di test generati
- Consente di ricostruire il modello della GUI a partire dagli output del processo

Obiettivi

- Si vuole rendere capace l'Android Ripper di:
  - Variare il valore dei sensori
  - Variare la posizione geografica del dispositivo
  - Simulare la ricezione di sms e chiamate
  - Interagire con la fotocamera
  - Gestire le reti wifi e dati

- Si vuole procedere
  - Utilizzando soluzioni esistenti dove possibile
  - Implementando nuove soluzioni dove necessario

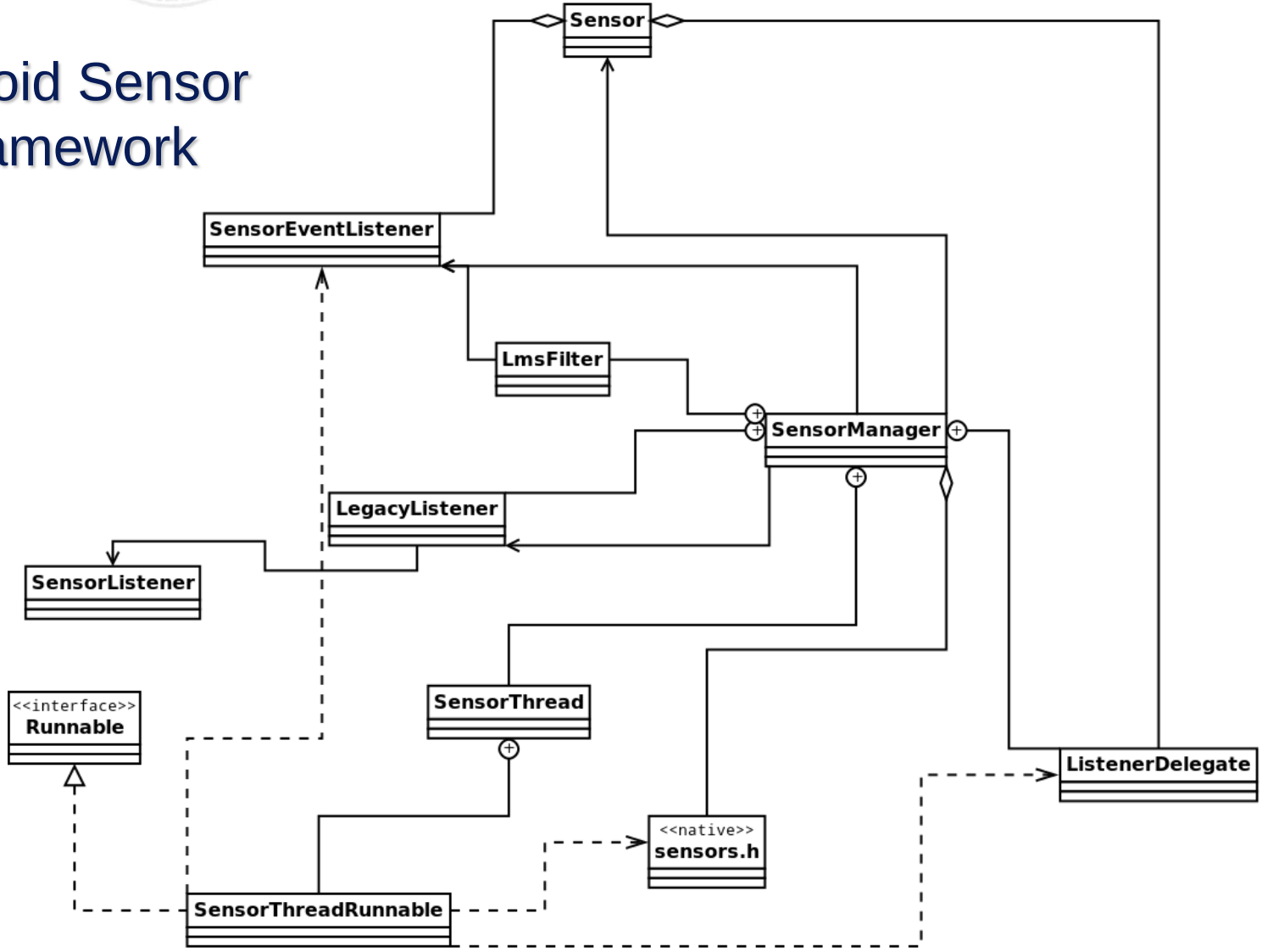
Problemi

- L'emulatore Android ha delle limitazioni
  - non c'è il supporto per la ricezione di chiamate reali
  - non c'è il supporto per la determinazione della connessione ad internet
  - non c'è supporto per il WiFi
  - non c'è il supporto per i sensori fisici
- Nel caso dei sensori fisici, le soluzioni esistenti sono difficilmente integrabili in Android Ripper
- È impossibile disconnettere l'emulatore dalla rete poiché necessaria per il funzionamento di DDMS
- Emulare una fotocamera a basso livello è oneroso

Soluzioni

- Sensori fisici
  - Implementazione di librerie basate sul Sensor Framework di Android
  - Instrumentazione del codice sorgente
- Variazione della posizione geografica del dispositivo (GPS)
  - Utilizzo delle funzionalità messe a disposizione dall'SDK
- Simulazione di ricezione chiamate ed SMS
  - Utilizzo di una socket connesso alla console DDMS dell'Emulatore
- Interazione con la fotocamera
  - Realizzazione di Activity che rispondano agli Intent al posto di quelle di sistema
- Gestire le reti WiFi e dati
  - Non facilmente realizzabile
  - Ininfluente sulla connessione dell'emulatore

Android Sensor Framework



Sperimentazione

- La sperimentazione è stata effettuata su
  - WordPress
    - Interazione con la fotocamera
    - GPS
  - Marine Compass (bussola)
    - Sensore di Orientazione
  - Bubble (livella)
    - Accelerometro
  - PedoMeter (contapassi)
    - Accelerometro
    - Sensore di Orientazione
  - OmniDroid
    - Ricezione di telefonate ed SMS

Risultati Sperimentazione

APPLICAZIONE	EVENTI CONSIDERATI	COPERTURA (LOC)	COPERTURA (METODI)
Marine Compass	• SENSORE DI ORIENTAZIONE	**97%** (92%) **463/475** (435/475)	**93%** (86%) **27/29** (25/29)
WordPress	• FOTOCAMERA • GPS	**46%** (45%) **4599,3/10007** (4505,2/10007)	**53%** (53%) **784/1479** (779/1479)
Bubble	• ACCELEROMETRO • SENSORE DI ORIENTAZIONE	**75%** (60%) **464,4/616** (371,1/614)	**74%** (66%) **85/115** (75/113)
Pedometer	• ACCELEROMETRO	**67%** (66%) **544,1/807** (528,2/805)	**72%** (72%) **161/225** (160/225)
Omnidroid	• TELEFONATE • SMS	**57%** (56%) **3480,2/6130** (3408,6/6130)	**60%** (58%) **813/1361** (789/1361)

- Generazione di nuovi eventi
- Incremento della copertura del codice sorgente

Conclusioni

- È stata aggiunta la possibilità di generare nuove tipologie di eventi nell'Android Ripper
- Questi nuovi eventi permettono di aumentare la copertura del codice sorgente
- Rilevazioni di eccezioni non gestite nel codice sorgente relativo a questi nuovi eventi

Sviluppi Futuri

- Simulare una variazione dei sensori secondo pattern più complessi (ad esempio shake)
- Simulare risposte ad Intent diretti verso altre applicazioni
- Rilevazione automatica dei sensori utilizzati
- Sviluppare tecniche euristiche per la valorizzazione degli input dei sensori
- Sviluppare tecniche per la generazioni di casi di test multi-threaded