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

tesi di laurea

Realizzazione di un Tool per l'iniezione automatica di difetti all'interno di codice Javascript

Anno Accademico 2009/2010

relatore
Ch.mo prof. Porfirio Tramontana

correlatore
Ch.mo ing. Domenico Amalfitano

candidato
Vincenzo Riccio
Matr. 534/2557

Contesto

Testing di Rich Internet Application (RIA) basate su AJAX

Stato dell'arte

Processi di Testing di RIA proposti dal DIS che utilizzano tracce d'esecuzione reali ottenute:

- Da sessioni utente
- Attraverso tecniche di crawling

Obiettivo

Supportare la sperimentazione riguardo l'efficacia dei processi di testing di RIA proposti

Soluzione proposta

Sviluppo di tecniche e strumenti per l'automatizzazione delle operazioni di iniezione di difetti in codice Javascript a supporto degli esperimenti condotti sul Testing di RIA

Processo di valutazione dell'efficacia del testing

Valutare la capacità delle tecniche di testing proposte di individuare malfunzionamenti in una RIA reale

necessita

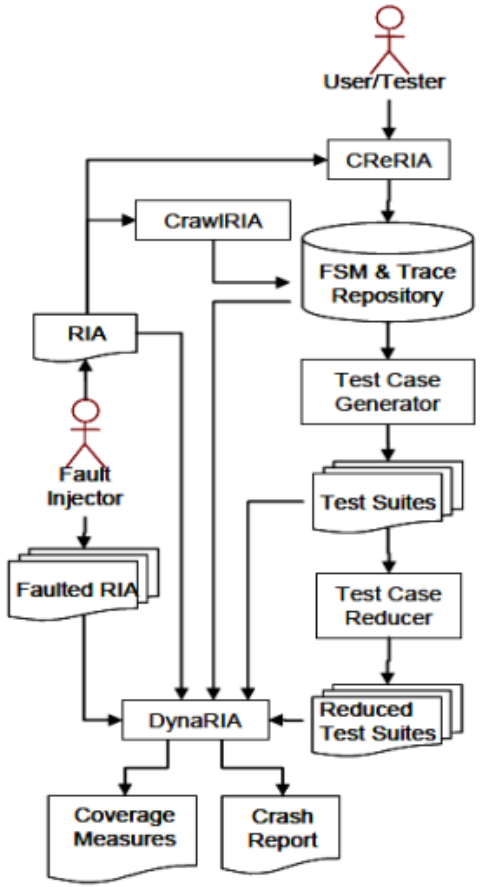
disponibilità del codice sorgente della RIA con difetti reali conosciuti

ma

difetti reali non sono sempre disponibili o conosciuti

Fault Seeding:

- **Step 1: Iniettare tipologie di difetti noti nel codice di una RIA**
- **Step 2: Rilevare malfunzionamenti sfruttando le tecniche e gli strumenti di testing proposti**



Modalità d'iniezione di difetti

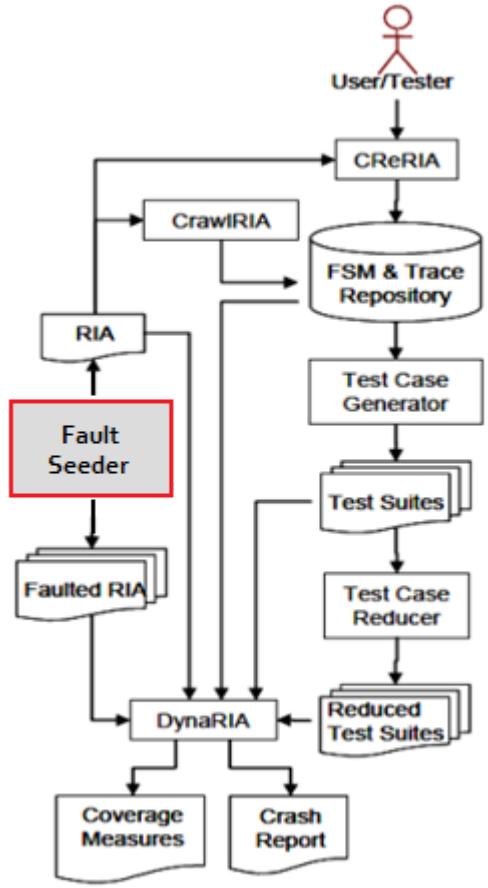
- Automatica
- Basata su tassonomia
- Tecnica *compile-time* basata su mutazioni del codice
- Sfrutta le espressioni regolari
- Codice JS statico del client

Vantaggi

- Possibilità di iniettare nel codice della RIA un maggior numero di difetti in minor tempo rispetto all'approccio "manuale"
- Possibilità di scegliere la distribuzione statistica delle tipologie di difetti

Contributo

- Sviluppo di una Tassonomia di difetti delle RIA basate su AJAX
- Sviluppo di un sistema software a supporto del Fault Seeding automatico



Sviluppo della Tassonomia dei difetti

Fase 1: Studio delle tassonomie presenti in letteratura

- Tassonomia dei difetti delle applicazioni Web di Marchetto et al.
- Tassonomia dei difetti delle GUI di Memon et al.
- Classificazione errori del linguaggio Javascript

Fase 2: Proposta di una tassonomia dei difetti delle RIA

- **Dominio:** RIA basate su AJAX
- **Dimensione:** Natura dell'errore nel codice Javascript { E. Semantici; E. Runtime }
- **Sottocategorie:** Classi di difetti utilizzate da Memon et al. per il seeding di difetti in GUI

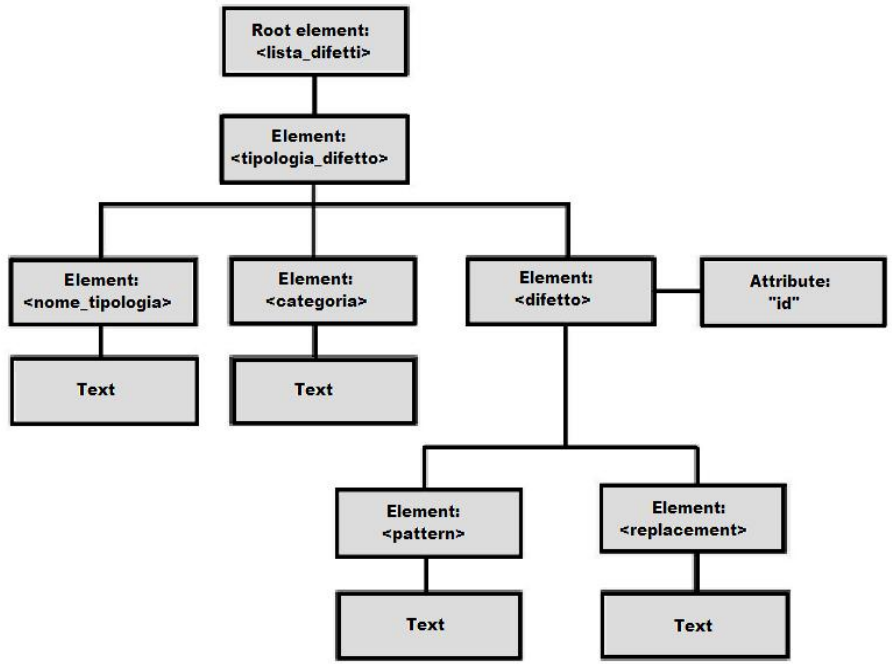
Estensione della Tassonomia

- Ciascuna tipologia di difetto è specializzata da un insieme di istanze
- Le istanze di difetti sono descritte da regole di mutazione del codice, composte da:
  - Pattern
  - Replacement

Espressioni regolari (*regexp*)
- Strumento compatto e flessibile per descrivere pattern complessi all'interno di testi
- Utilizzate per la ricerca e la sostituzione di porzioni di testo descritte dai pattern

Implementazione

La tassonomia è stata implementata come un documento XML
- Accessibile dal tool tramite XML DOM
- Consultabile e modificabile dal tester



eXtensible Markup Language
- Linguaggio di markup per la memorizzazione ed il trasporto dei dati
- Utilizzato per strutturare i dati attraverso tag definite dall'utente

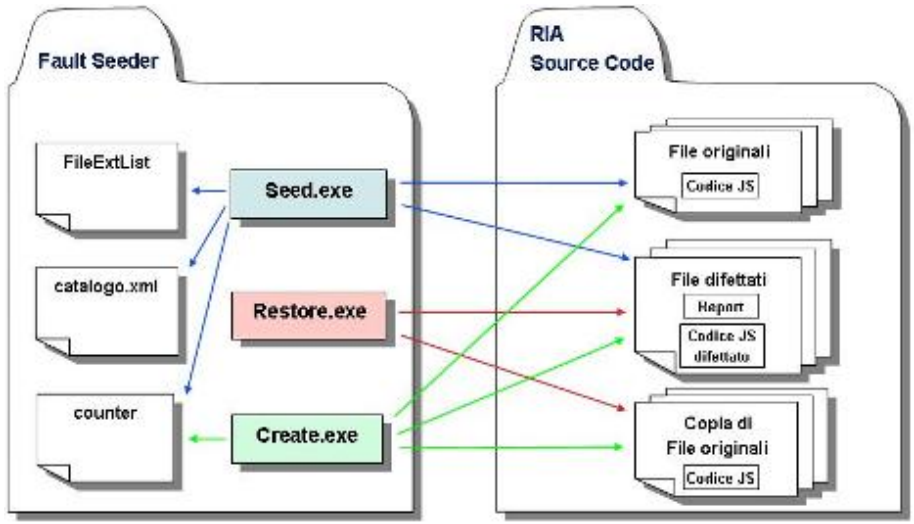
Fault Seeder

Sviluppato in python

Funzionalità:

- **Iniezione automatizzata di difetti nei file delle RIA contenenti codice JS secondo diverse modalità:**
  - Iniezione di un difetto D in un file F
  - Iniezione di un difetto D in un file casuale contenuto in una cartella
  - Iniezione di K difetti casuali in un file F
  - Iniezione di K difetti casuali in una cartella
  - Creazione di un set di N difetti
- **Creazione di copie difettate della RIA**
- **Ripristino della versione originale della RIA**

Architettura



Le tre funzionalità principali sono state implementate in tre eseguibili per garantire maggiore flessibilità nel processo di Fault Seeding

python : py2exe

Fair Seeding

Funzionalità del Tool che consente d'iniettare un set d'istanze di tipologie di difetti distribuite in modo proporzionale ai punti del codice in cui è possibile iniettare tali istanze (opportunità)

L'utente fornisce in input:

- **Cardinalità del set**
- **Seme di casualità**
- **Path della cartella in cui si trova il codice della RIA**

Il sistema:

- **Ispeziona il codice per ottenere per ciascuna classe di difetti le opportunità d'iniezione**
- **Distribuisce equamente (fair) le istanze per ciascuna classe di difetti in base alla regola:**

$$\varphi_i \approx ((\omega_i / \Omega) \cdot \Phi)$$

Ove:

- φ_i istanze di difetti iniettate per classe i
- Ω somma di tutte le opportunità per tutte le classi
- ω_i numero di opportunità per classe
- Φ numero totale di difetti da iniettare

Sperimentazione

RIA target Tudu Lists — Getting Things Done!

- Gestisce liste di cose da fare
- AJAX-based
- Open source

Goal

Dimostrare l'efficacia del Tool in uno scenario reale

Modalità di sperimentazione

- **Iniezione nei file del codice con estensione .jsp**
- **Creazione di un Set di file difettati di cardinalità 100**
- **Seme di casualità 0**

Risultato dell'esperimento

- **Ω=546**
- **105 difetti iniettati (≈Φ)**
- **Le regole di mutazione sono valide ed efficaci**

CATEGORIA	TIPOLOGIA DI DIFETTO	ID	ω_i	φ_i
SEMANTIC	Modifica di un operatore relazionale	002	2	0
		007	8	2
		008	8	2
		009	8	2
		010	5	1
		011	5	1
		012	5	1
		035	5	1
	Inversione dello statement	036	25	5
	Modifica di operatori logici	038	10	2
	Variazione dell'ordine dei parametri	040	37	7
	Set/Return di un valore stringa differente	041	48	9
		042	5	1
		043	73	14
	Set/Return di un valore intero differente	044	4	1
		045	2	0
		046	2	0
	Set/Return di un valore booleano differente	048	12	3
		049	2	0
		050	1	0
RUNTIME	Invocazione di un metodo sintatticamente simile ma non definito	051	87	16
	Set/return di una variabile sintatticamente simile ma non definita	052	49	9
		053	43	8
	Set/Return di un attributo differente	057	40	8
		058	57	11
		059	3	1

Esempio

```
// Delete the current todo list.
function deleteTodoList(listId) {
 hideTodosLayers();
 var listId = document.forms.todoForm.listId.value;
 if (listId != null && listId != "null" &&  listId != "") {
  var sure = confirm("<fmt:message key="todo.lists.delete.confirm"/>");
  if (sure) {
   dwr.engine.beginBatch();
   todo_lists.deleteTodoList(listId);
   todos.forceGetCurrentTodoLists(replyCurrentTodoLists);
   document.forms.todoForm.listId.value = null;
   dwr.util.setValue('todosTable',
    "<div class='message'><fmt:message key="todo.lists.delete.ok"/></div>");
   dwr.engine.endBatch();
   tracker('/ajax/deleteTodoList');
  }
```

```
if (!(sure)) {
```

- todos_menu.jsp
- todos_menu.jsp.fault0015

Regola mutazione

- **Pattern:** (\bif\b\s*\(\s*)(!?.+\))
- **Replacement:** \1!(\2)
- **Inversione dello statement in una condizione if**

```
<%--

File difettato
File originale: C:\...\todos_menu.jsp
Categoria difetto: semantic
Tipologia difetto: Inversione dello statement
ID#036
Mutazione effettuata alla 10^ occorrenza del
     pattern
-: if (sure)
+: if (!(sure))

--%>
```

Obiettivi raggiunti

- È stata sviluppata una tassonomia di difetti iniettabili automaticamente nel codice client delle RIA AJAX-based
- È stato realizzato un sistema software che consente di:
  - Iniettare difetti in file del codice sorgente di RIA
  - Creare copie difettate di RIA
- È stata dimostrata l'efficacia del software attraverso un esperimento su una RIA reale: *Tudu Lists*

Conclusioni

- Sono stati riscontrati limiti nelle potenzialità espressive delle regexp:
  - Semplicità dei pattern, che interessano singole linee di codice
  - Alcune classi di difetti individuate non sono state implementate nella tassonomia
- Il software prodotto implementa appieno le funzionalità richieste

Sviluppi futuri

⇨
- Studiare la fattibilità di un approccio all'ispezione del codice basato su parsing
- Analizzare diverse RIA reali per implementare la creazione di set di difetti distribuiti con criterio statistico

⇨
- "Agganciare" il Fault Seeder realizzato ai sistemi di Testing in fase di sviluppo