AUTOMATIC SOFTWARE REPAIR

MAROUANE, HECTOR, QUENTIN

Articles

- Automatic Software Repair: a Bibliography (Martin Monperrus)
- Current challenges in automatic software repair (Claire Le Goues, Stephanie Forrest & Westley Weimer)
- GenProg: A Generic Method for Automatic Software Repair (Claire Le Goues; ThanhVu Nguyen; Stephanie Forrest; Westley Weimer)
- Automatic Program Repair as Semantic Suggestions: An Empirical Study (Diogo Campos; André Restivo; Hugo Sereno Ferreira; Afonso Ramos)
- Advancements in Self-Healing Technology for Software Systems (Sreedhar Reddy Konda)
- APR4Vul: an empirical study of automatic program repair techniques on real-world Java vulnerabilities (Quang-Cuong Bui.)

C’est quoi?

Bugs
Security Vulnerabilities
Complexity

Automatic Software Repair

- **Article :** Current challenges in automatic software repair, Published: 07 June 2013

Problématique

“Quelles sont les technologies utilisées pour la réparation automatique de logiciels**?**”

Sommaire

01

Behavioral Repair

- Explication de la méthode
- Cartographie des outils

02

State Repair

- Explication de la méthode
- Cartographie des outils

03

Ouverture

- Perspective d'avenir

01 Behavioral Repair

Le principe

Modifier le comportement du programme en réparation

Exemple

```
1  char* ProcessRequest() {
2   ...
3   while(l=sgets(l,sock)) {
4    if(l=="Request:")
5     strcpy(req_type,l+12)
6    if(l=="Content-Length:")
7     len=atoi(l+16);
8   }
9   if(req_type=="GET")
10   buff=DoGETReq(sock,len);
11  if(req_type=="POST") {
12   sz=sizeof(char);
13   buff=calloc(len,sz);
14   rc=recv(sock,buff,len)
15   buff[len]='\0';
16  }
17  return buff;
18 }
```

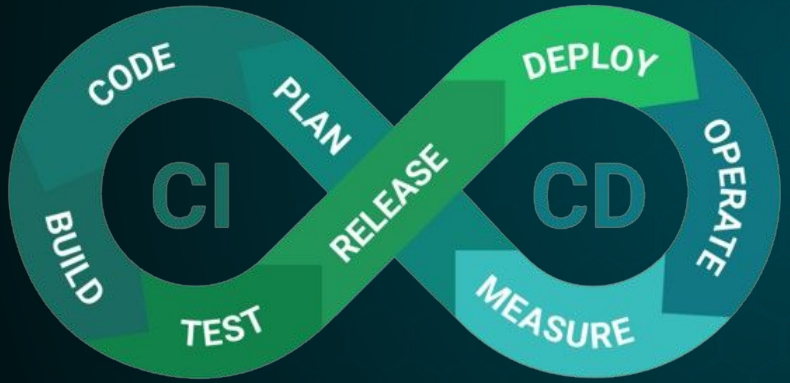
(a) Webserver code snippet.

```
3  ...
4    if(l=="Request:")
5     strcpy(req_type,l+12)
6    if(l=="Content-Length:")
7     len=atoi(l+16);
8   }
9   if(req_type=="GET")
10   buff=DoGETReq(sock,len);
11  if(req_type=="POST") {
12+   if (len <= 0)
13+    return null;
14   sz=sizeof(char);
15   buff=calloc(len,sz);
16   rc=recv(sock,buff,len)
17   buff[len]='\0';
18  }
19  return buff;
20 }
```

(b) Patched webserver.

- **Article :** Current challenges in automatic software repair

Quand exécuter un tel programme ?



- Exécution manuelle
- IDE
- CI

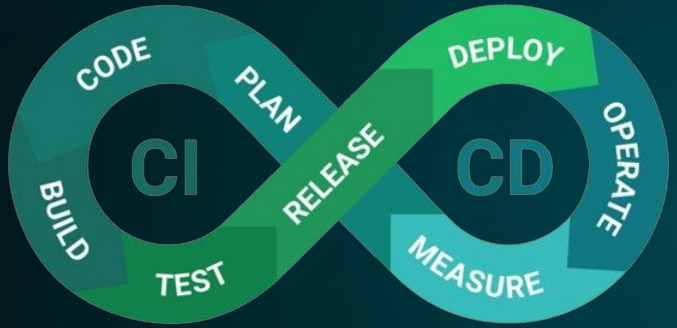
Les technologies actuelles

- GenProg : language C
  https://github.com/squaresLab/genprog-code
- Astor : language Java (créé en partie par l'Inria)
  https://github.com/SpoonLabs/Astor
- Nopol : language Java
  https://github.com/SpoonLabs/nopol
- ARJA : language Java
  https://github.com/yyxhdy/arja

02 State Repair

Modifier l'état du programme en réparation

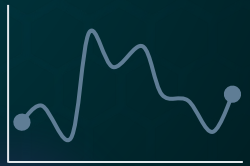
Le Principe



- Forcément durant l'exécution
- Modification de l'input, le heap, la stack, l'environnement ...
- Nécessite un oracle disponible en production

Contrat non-fonctionnel

Contrat fonctionnel



Contrat déduit

Les opérateurs de réparation

Reboot et
Micro-reboot

Alternatives

Roll back et
Checkpoint

Reconfiguration

- **Article :** Automatic Software Repair: a Bibliography

Les opérateurs de réparation

Input
Modification

Roll Forward

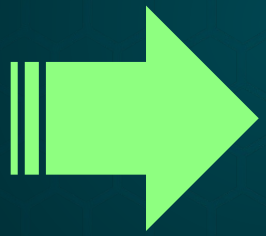
Environnement
Perturbations

Collaborative Repair

- **Article :** Automatic Software Repair: a Bibliography

Bilan

- Restart
- Checkpoint/Snapshot
- Alternative implementation
- Input modification
- Environment modification
- Invariant restoration
- Error virtualization
- Collaborative repair



- Dira (Récupérer après un hijack)
- Hooks to a recovery (Créer des objets par défauts pour remplacer les valeurs NULL)
- Vigilante (Filtre et modifie les inputs du buffer)
- Jolt (Détecter et échapper à une boucle infinie en monitorant la mémoire)
- ClearView (Restauration à partir d'invariants appris)
- Assure (Virtualisation d'erreur)

- **Article :** Automatic Software Repair: a Bibliography

Ouverture 03

Automatic Software Repair

Une avancée majeure pour atténuer les risques associés aux bugs

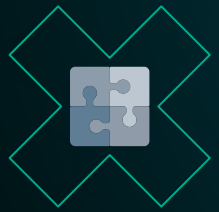
- La réparation comportementale cible la modification du code du programme
- La réparation d'état s'attaque à la modification de l'état d'exécution en temps réel

Des défis subsistent, notamment en termes de complexité croissante des environnements logiciels et de scalabilité. Les coûts et les ressources nécessaires, ainsi que les limitations dans des contextes logiciels complexes, restent des aspects à considérer.

Perspective d’avenir

IA

- Modèles prédictifs basés sur l'IA



DL

- Adaptabilité face à de nouvelles situations et demandes opérationnelles.

- **Article :** Advancements in Self-Healing Technology for Software Systems, January 2024

MERCI!