AFL GUI tool

Homework 7

112598010 林建澄
112598052 林祐清

目錄

1. Requirement Document..... 3
1.1. Change History..... 3
1.2. Problem statement.....4
1.3. Summary of System Features..... 4
1.4. Use Case Diagram.....4
1.5. Use Cases..... 5
1.6. Non-functional Requirements and Constraints..... 9
1.7. Glossary.....9
1.8. Software Environments.....9
2. Domain Model..... 10
2.1. Domain class diagram showing only concepts..... 10
2.1.1. Classes Identified.....10
2.1.2. Bad Classes.....10
2.1.3. Good Classes..... 10
2.2. Add associations..... 11
2.3. Add attributes.....12
3. Design..... 13
3.1. Logical architecture.....13
3.2. Use-Case Realizations with GRASP Patterns..... 13
3.2.1. System Sequence Diagram (SSD)..... 13
3.2.2. Sequence Diagram..... 15
3.3. Design class model.....18
4. Implementation Class Model.....19
4.1. Implementation class model..... 19
4.2. difference between implementation class model and design class model..... 19
4.2.1. Comparison with design and implementation class... 19

4.2.2. Summary of implementation class/method changed. 21
4.3. Calculate Line of Code.....22
5. Programming..... 23
5.1. Snapshots of system execution..... 23
5.2. Source Code Listing (note: select and list the most important source code;no more than 10 pages)..... 27
6. Unit Testing..... 33
6.1. Snapshots of testing result.....33
6.2. Unit Testing Code Listing..... 34
- Measurement.....41

1. Requirement Document

1.1. Change History

Revision	Description	Data
1	Problem Statement	2024-02-28
2	Change History Problem Statement refinement Summary of System Features Use Case Diagram Use Cases Non-functional Requirements and Constraints Glossary Software Environments	2024-03-13
3	Revise requirement document Domain model Domain class diagram showing only concepts Add associations Add attributes	2024-03-27
4	Revise requirement document and domain model Logic architecture Use-Case Realizations with GRASP Patterns System Sequence Diagram (SSD) Sequence Diagram Design Class Model	2024-04-17
5	Revise requirement document and domain model Implementation Class Model Draw an implementation class diagram for your system Show the difference between implementation class model and design class model	2024-05-01
6	Run the same steps through homework 2 to homework 4. Revise the requirement Document Revise the domain model	2024-05-22
7	Revise the implementation class diagram Revise the difference between implementation class model and design class model	2024-06-13

1.2. Problem statement

AFL ++ 是一種廣泛使用的 fuzzing 工具，可有效的發現軟體中的漏洞。然而，AFL ++ 的原生介面較為簡陋，使用上較不直覺。為了提升 AFL ++ 的使用體驗，我們計畫開發一款 AFL ++ 專用的 GUI 工具。

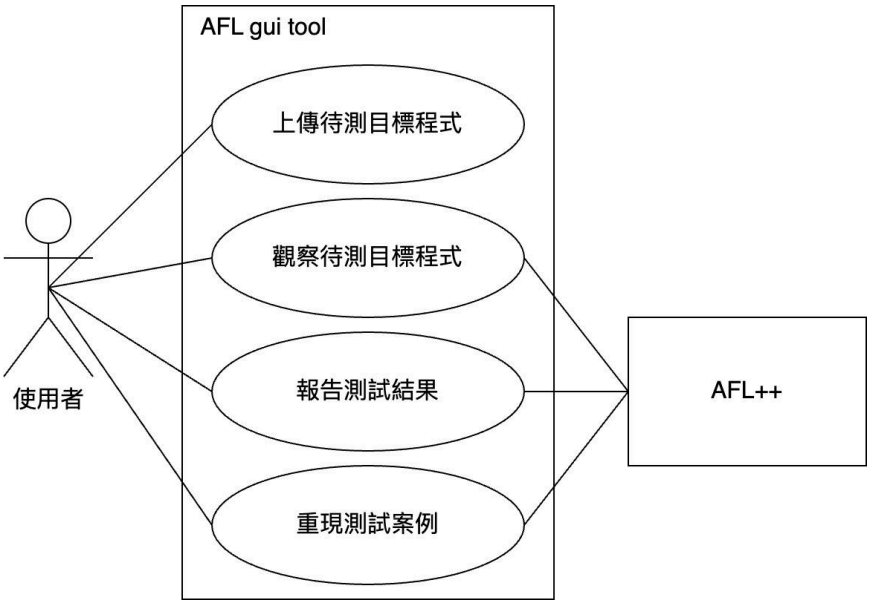
這款工具具有以下功能：

- 瀏覽 **AFL ++** 執行狀態及結果：使用者可以透過 GUI 工具即時查看 AFL ++ 的執行狀態，包括已測試的輸入數量、發現的 crash 數量等。此外，使用者也可以使用 GUI 工具瀏覽 AFL ++ 找到的 crash 資訊，包括 crash 的追蹤等。
- 以圖表呈現 **crash** 資訊：GUI 工具可以將 crash 資訊以圖表的方式呈現，讓使用者更容易理解 crash 的原因。例如，GUI 工具可以顯示 crash 發生時程式呼叫堆疊的圖表，讓使用者快速找出 crash 的源頭。
- 圖形介面上傳待測目標程式：GUI 工具提供圖形介面，讓使用者可以輕鬆上傳待測目標程式。使用者只需將待測目標程式的檔案拖曳到 GUI 工具即可。

1.3. Summary of System Features

Feature ID	Description
FEA-01	使用者可以透過網頁 GUI 介面上傳和執行想要測試的目標，而不必透過終端視窗下命令。
FEA-02	使用者可以透過網頁 GUI 介面即時觀察當前模糊測試執行的狀態。
FEA-03	使用者可以透過網頁 GUI 介面觀察造成 crash 的輸入。
FEA-04	使用者可以透過網頁 GUI 介面重新執行造成 crash 的輸入。

1.4. Use Case Diagram



1.5. Use Cases

Use case 1	Comment
Use Case Name	上傳待測目標程式
Scope	AFL GUI tool
Level	使用者界面操作
Primary actor	使用者
Stakeholders and Interests	使用者希望輕易地上傳待測程式，系統需要有效地處理上傳的檔案並準備進行 fuzzing
Preconditions	1. 使用者已編譯待測程式成 binary 執行檔案，並準備上傳 2. 使用者開啟 GUI 工具
Success Guarantee	待測程式成功上傳至 AFL GUI tool 且準備被 fuzzing
Main success scenario	1. 使用者進入上傳頁面 2. 系統向使用者顯示資訊(包括上傳檔案、參數設定) 3. 使用者選擇要上傳的待測程式 4. 使用者設定執行時要使用的相關參數 5. 使用者確認上傳操作 6. GUI 工具接收並檢查檔案和參數 7. 系統準備開始執行 fuzzing 8. 系統向使用者顯示上傳結果和使用的參數
Extensions	● 6a. 上傳檔案格式錯誤 1. 顯示錯誤訊息並要求使用者重新上傳有效的檔案 2. 使用者重新上傳待測檔案 3. GUI 工具重新檢查待測檔案格式
Special requirement	None
Technology and Data Variations List	None
Frequency of Occurrence	Often

Use case 2	Comment
Use Case Name	觀察待測目標程式
Scope	AFL GUI tool
Level	使用者界面操作
Primary actor	使用者
Stakeholders and Interests	使用者希望即時觀察 fuzzing 進度，包括測試輸入數量、crash 數量，當前執行狀態等訊息
Preconditions	已成功上傳待測目標程式且 fuzzing 進行中
Success Guarantee	使用者能夠清楚地看到目前 fuzzing 的進展，包括各類重要的統計訊息
Main success scenario	1. 使用者在 GUI 工具中選擇"查看狀態"頁面 2. 選擇想要查看的待測程式 3. 系統顯示待測程式的 fuzzing 進度和當前狀態 4. 使用者可以轉換想要執行的 fuzz target 5. 系統顯示轉換後狀態 6. 使用者閱讀並瞭解狀態

Extensions	● 2a. 無可查看的待測程式
	  1. 使用者尚未上傳任何待測程式
	● 2b. 待測程式並非執行中
	  1. 系統顯示待測程式未在執行中
	  2. 使用者重新選擇待測程式
	● 3a. 系統出現無法顯示的異常：
	  1. 系統顯示錯誤資訊
	  2. 使用者可選擇重新整理
	● 4a. 轉換執行程式已在運行中
	  1. 系統顯示錯誤資訊
	  2. 回到3

Special requirement	即時更新機制、GUI 清晰的統計訊息展示
Technology and Data Variations List	None
Frequency of Occurrence	Sometimes

Use case 3	Comment
Use Case Name	報告測試結果
Scope	AFL GUI tool
Level	使用者界面操作
Primary actor	使用者
Stakeholders and Interests	使用者想獲得詳盡的 fuzzing 測試結果報告，系統需要提供足夠的資訊給使用者分析
Preconditions	至少完成一次 fuzzing 的測試，並有結果可供報告
Success Guarantee	測試結果被完整地報告出來，並以圖形介面呈現測試資訊
Main success scenario	1. 使用者進入"測試結果"頁面 2. 系統顯示所有待測程式 3. 使用者選擇要查看的待測程式的測試報告 4. 系統提取相關測試數據並生成報告 5. 系統展示測試報告，包括發現的漏洞、crash 資訊等 6. 系統以圖表呈現測試結果 7. 使用者評估測試報告

Extensions	● 4a. 測試報告不存在
	  1. 回到步驟3
	● 6a. 圖表無法顯示
	  1. 系統顯示錯誤原因
	  2. 回到步驟3
	● 7a. 遇到錯誤或無法生成報告
	  1. 系統提示錯誤並提供可能的解決方案
	  2. 回到步驟3

Special requirement	測試結果報告功能需能自動整理資料
Technology and Data Variations List	None
Frequency of Occurrence	Sometimes

Use case 4	Comment
Use Case Name	重現測試案例
Scope	AFL GUI tool
Level	使用者界面操作
Primary actor	使用者
Stakeholders and Interests	使用者希望能重現發生問題的測試過程，以便進一步的手動分析和除錯
Preconditions	已發現至少一個 crash 或異常，且系統已記錄對應的測試案例
Success Guarantee	測試案例能被準確地重播，而且使用者能夠觀察到重播結果和造成 crash 的原因
Main success scenario	1. 使用者進入”重現測試”的頁面 2. 系統顯示所有執行過的測試程式 3. 使用者從中選擇一個待測程式的一個 crash 案例 4. 使用者指示 GUI 工具進行重播 5. GUI 工具執行導致 crash 的測試資料 6. 系統將執行結果輸出到頁面 7. 使用者決定是否把結果記錄下來 8. 使用者分析重播結果
Extensions	● 3a. 待測程式沒有 crash 案例 1. 系統中沒有任何測試程式的 crash 案例 2. 系統顯示訊息，說明目前沒有可用的 crash 案例可供重現 3. 回到步驟2 ● 5a. 載入測試案例失敗 1. 嘗試載入所選擇的 crash 案例時發生錯誤 2. 系統顯示錯誤訊息，說明無法載入案例的原因(如檔案損毀或格式錯誤) 3. 回到步驟3
Special requirement	需要能夠準確記錄與再現 fuzzing 的輸入，並且涉及執行環境的參數配置信息
Technology and Data Variations List	None
Frequency of Occurrence	Sometimes

1.6. Non-functional Requirements and Constraints

NFR ID	Category	Description
NFR-01	Usability	使用者可輕易的使用 AFL GUI Tool 來執行及管理模糊測試。
NFR-02	Reliability	AFL GUI Tool 能夠正確的執行並顯示模糊測試的執行狀態。

1.7. Glossary

Item	Definition or Description
Fuzz testing	Fuzz testing (模糊測試)是一種軟體測試技術，透過向待測程式輸入大量隨機輸入，以發現軟體中的錯誤或是漏洞。
Fuzz target	Fuzz target 是模糊測試的待測目標，可以是一個軟體、系統或是可執行檔。
AFL ++	AFL ++ 是一個灰箱模糊測試工具，其透過程式碼覆蓋率來反饋並生成輸入，並且監控軟體的執行狀況，當軟體出現異常時，AFL ++ 會記錄下輸入，以便開發人員進行分析。 AFL ++ 是 AFL 的後續社群版本，提供更強大的輸入生成能力及性能。

1.8. Software Environments

This project will be written in Python and JavaScript programming language.

2. Domain Model

2.1. Domain class diagram showing only concepts

2.1.1. Classes Identified

User	FuzzTarget	FuzzResult	Crash	FuzzConfig
FuzzProgress	FuzzStatus	FuzzData		

- User: 使用者
- FuzzTarget: 待測程式
- FuzzResult: 模糊測試開始執行後產生的結果
- Crash: 待測程式執行過程中產生的崩潰
- FuzzConfig: 待測程式上傳時的相關參數設定
- FuzzProgres: 待測程式執行中要傳給UI顯示的相關資訊
- FuzzStatus: 待測程式測試結束後的所有狀態紀錄
- FuzzData: 每隔一段時間的待測程式狀態紀錄的集合

2.1.2. Bad Classes

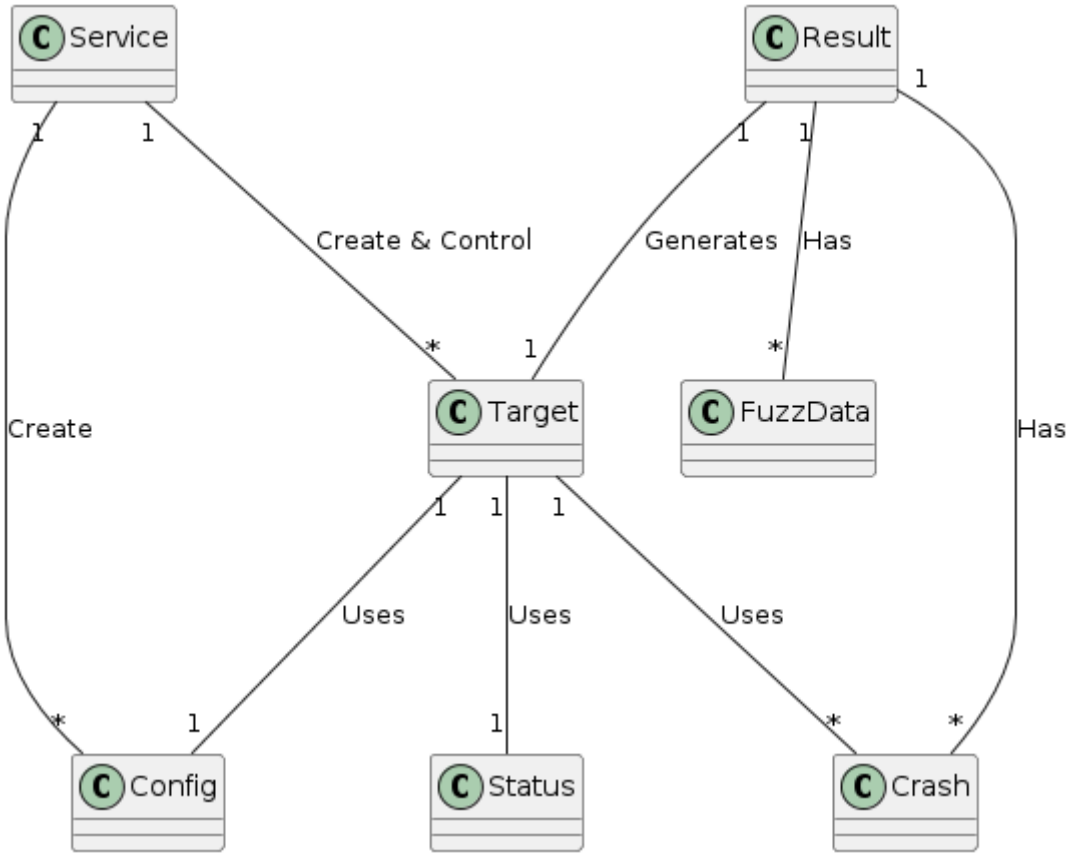
User				

- User: 因為使用這個工具的人只會有一個，不需要有User這個class去記錄相關資訊

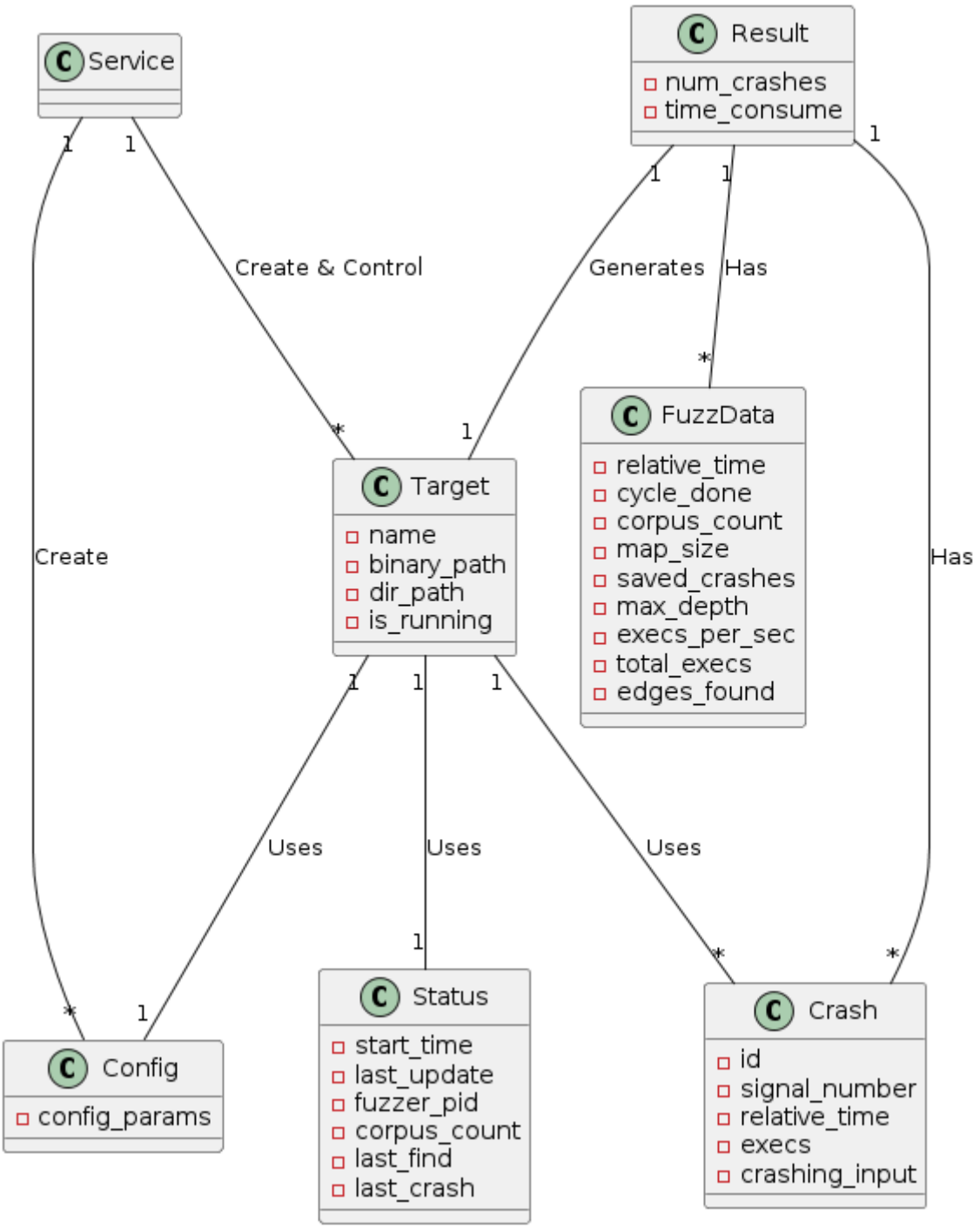
2.1.3. Good Classes

FuzzTarget	FuzzResult	Crash	FuzzConfig	FuzzProgress
FuzzStatus	FuzzData			

2.2. Add associations

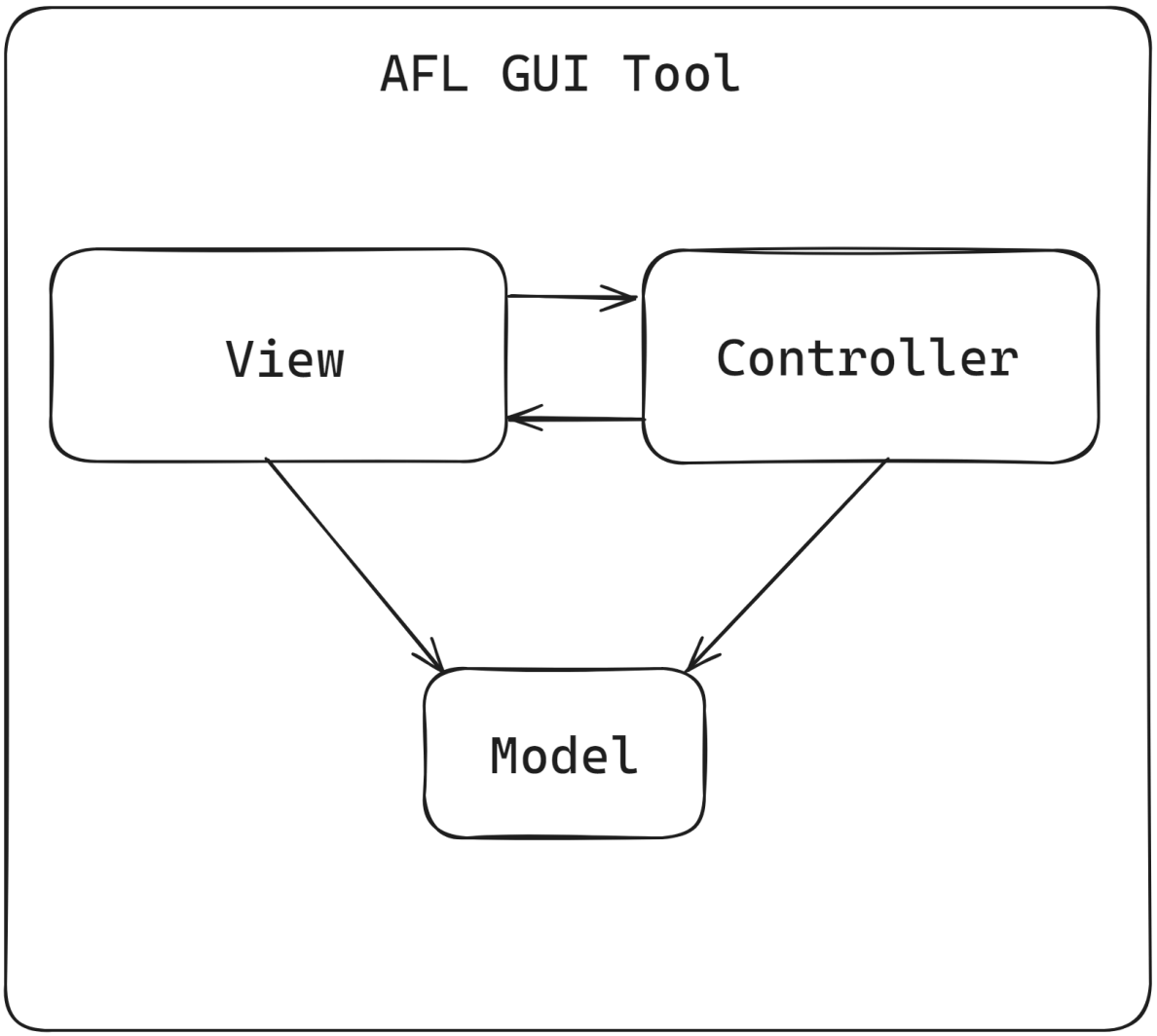


2.3. Add attributes



3. Design

3.1. Logical architecture



3.2. Use-Case Realizations with GRASP Patterns

3.2.1. System Sequence Diagram (SSD)

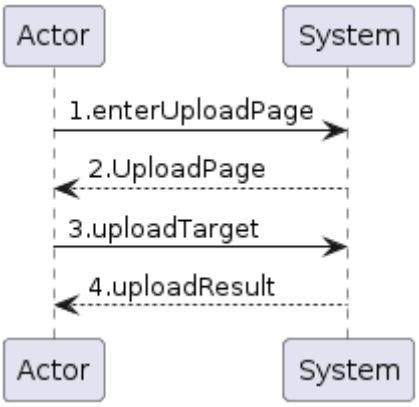


Figure 3.2.1-1. SSD of use case: 上傳待測程式

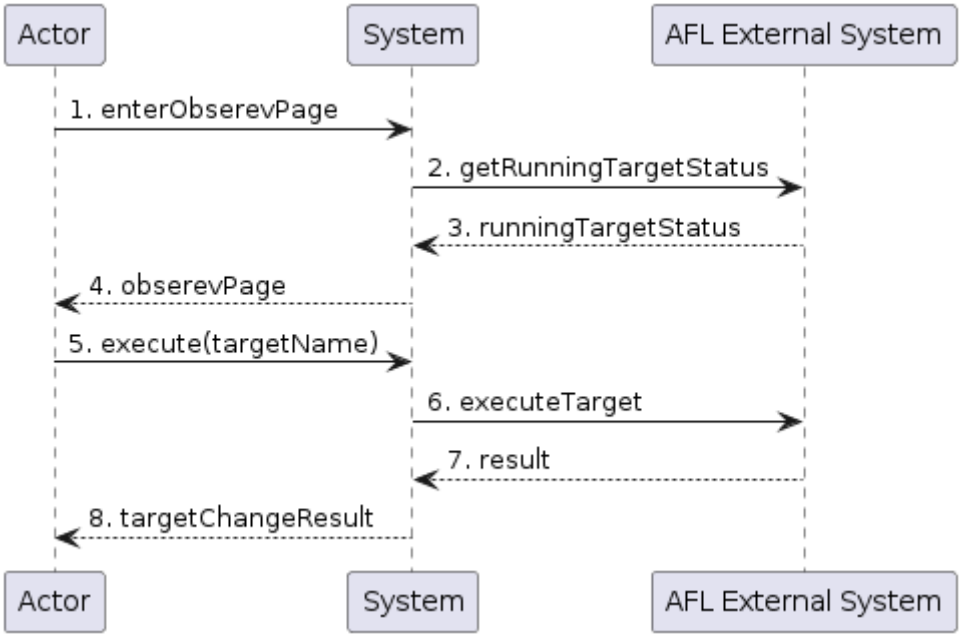


Figure 3.2.1-2. SSD of use case: 觀察待測目標

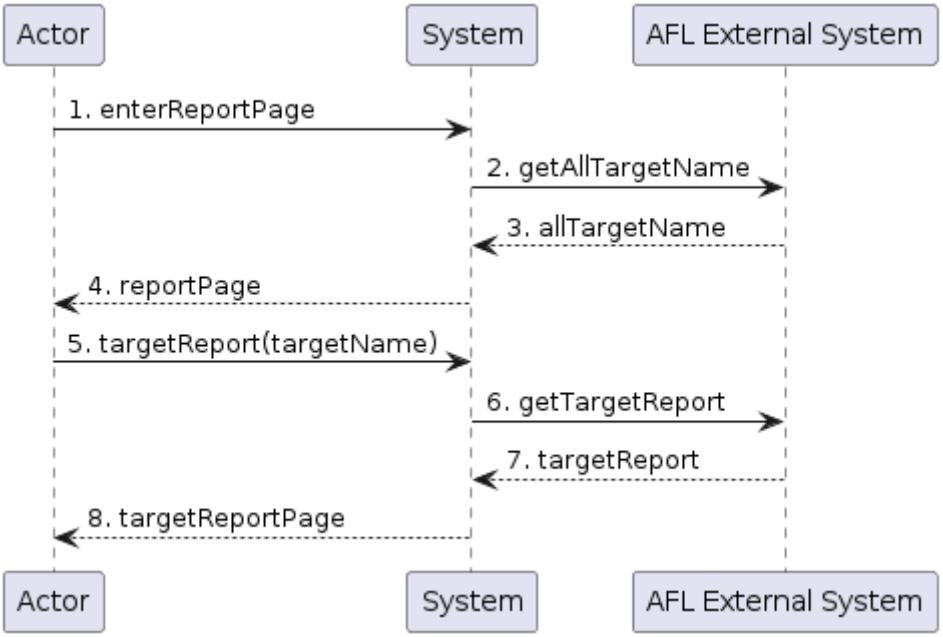


Figure 3.2.1-3. SSD of use case: 報告測試結果

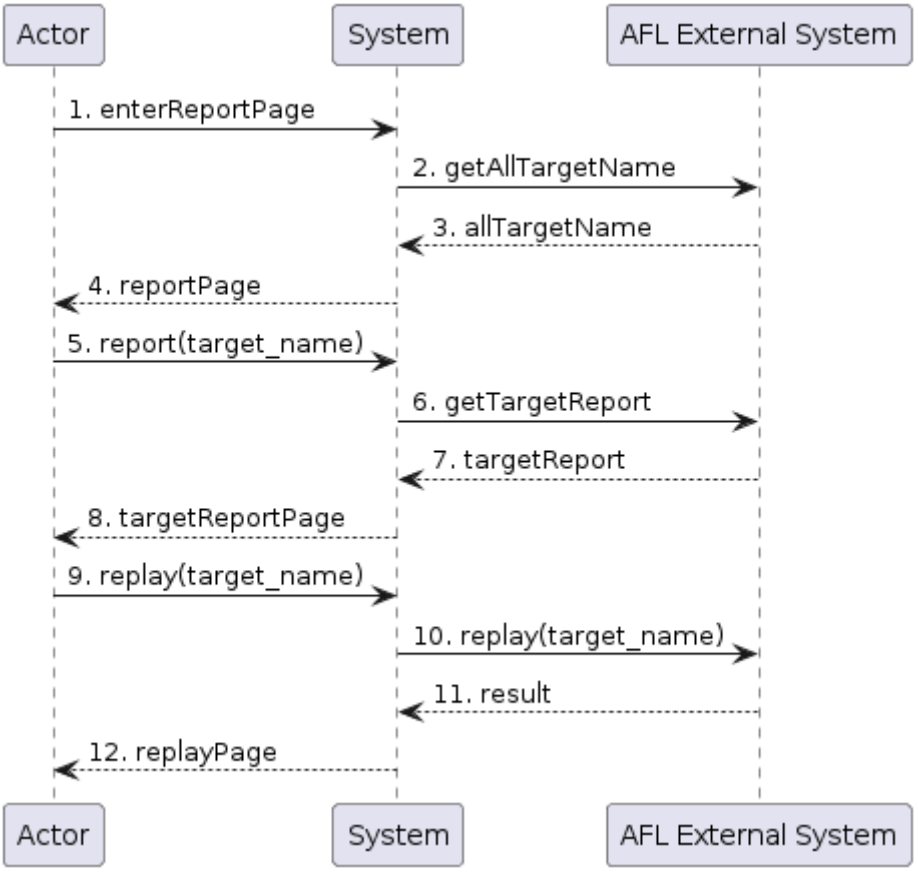


Figure 3.2.1-4. SSD of use case: 重現測試案例

3.2.2. Sequence Diagram

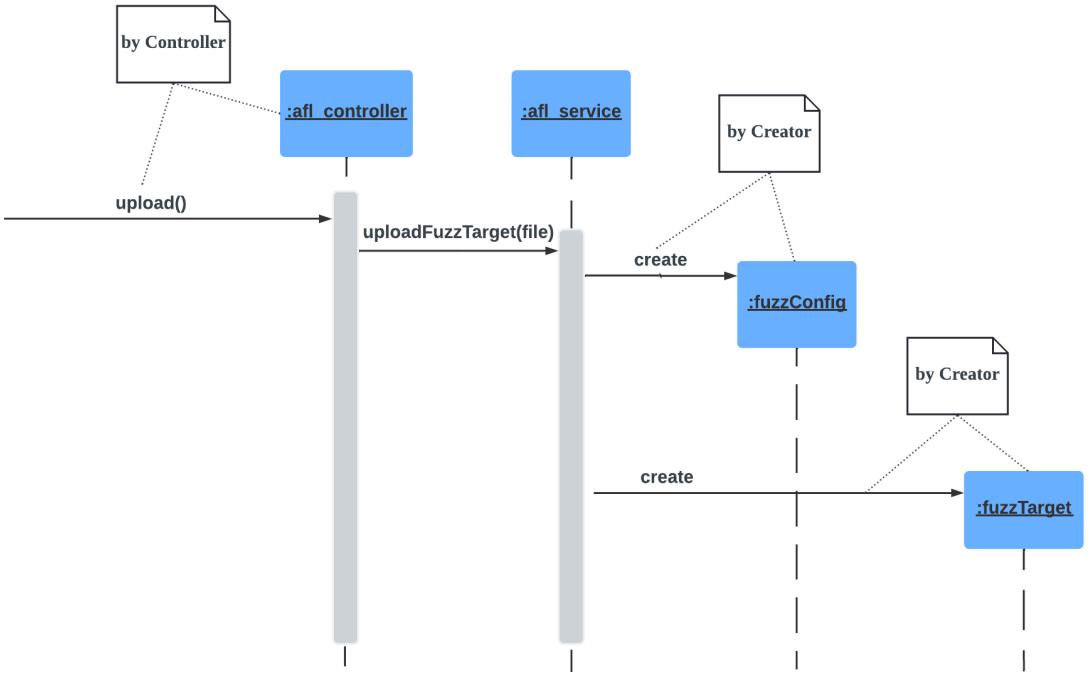


Figure 3.2.2-1. uploadFuzzTarget

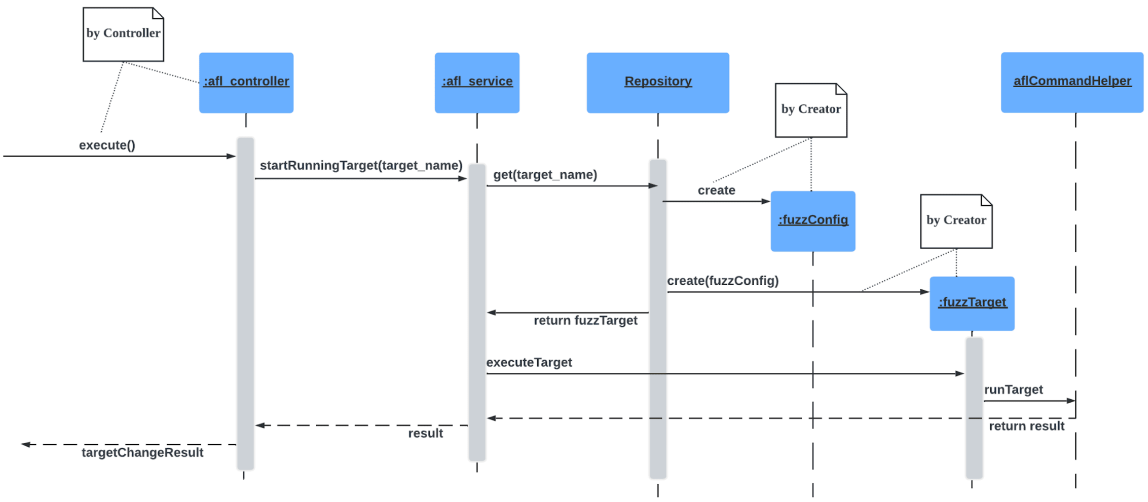


Figure 3.2.2-2. executeTarget

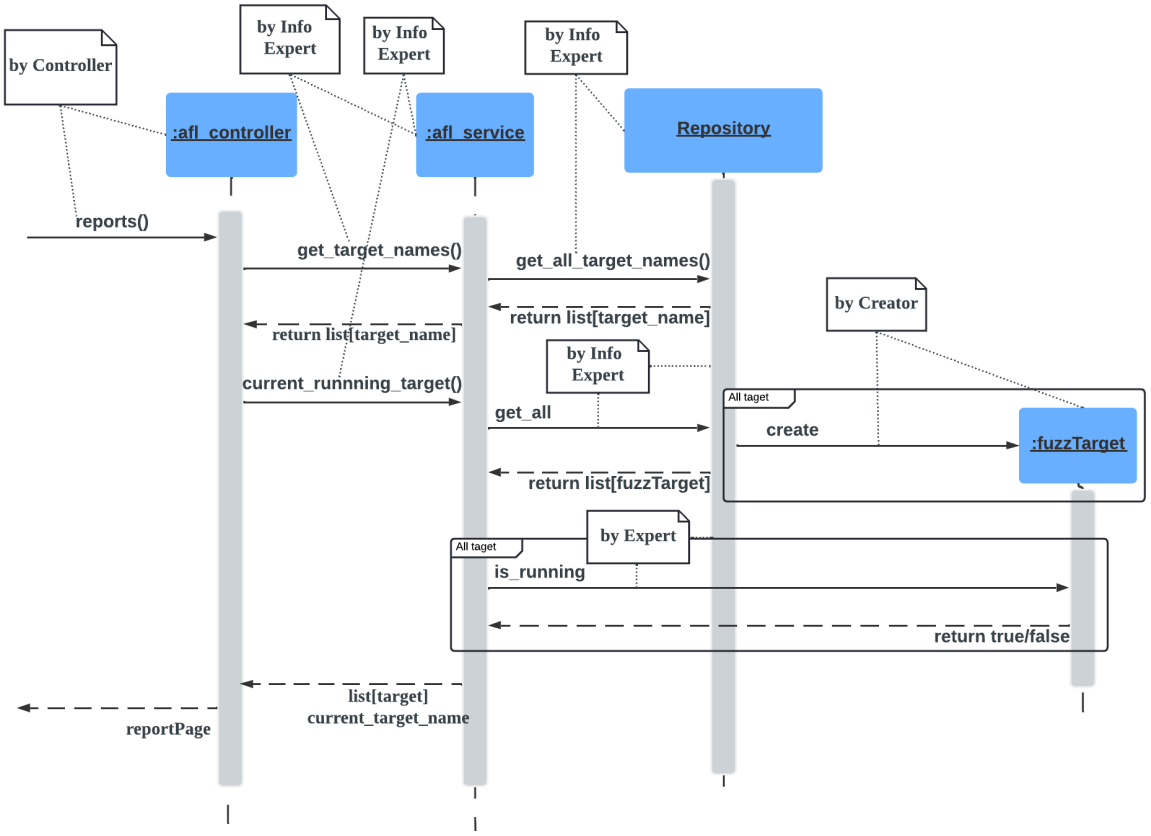


Figure 3.2.2-3. report

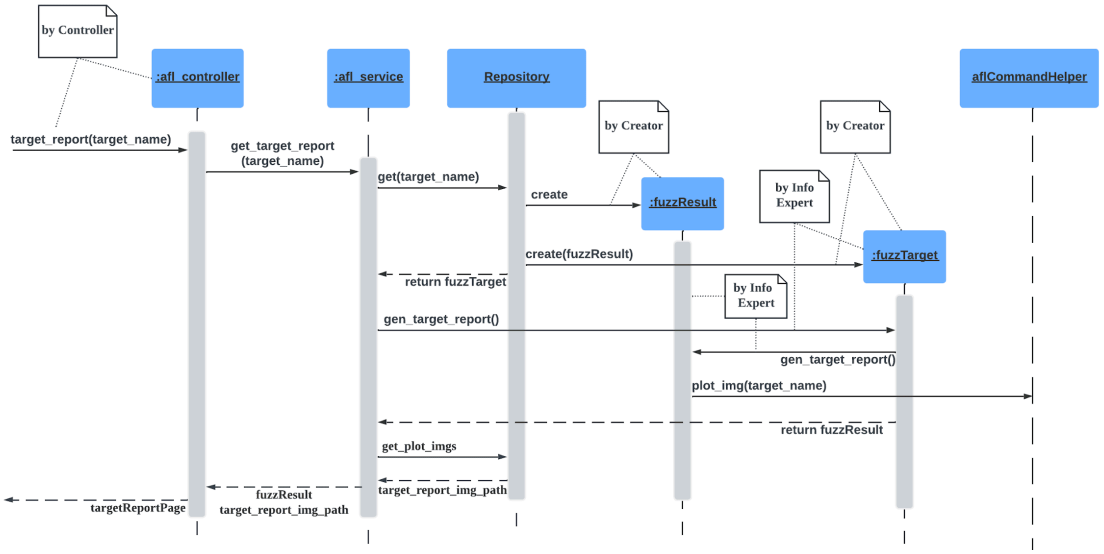


Figure 3.2.2-4. target_report

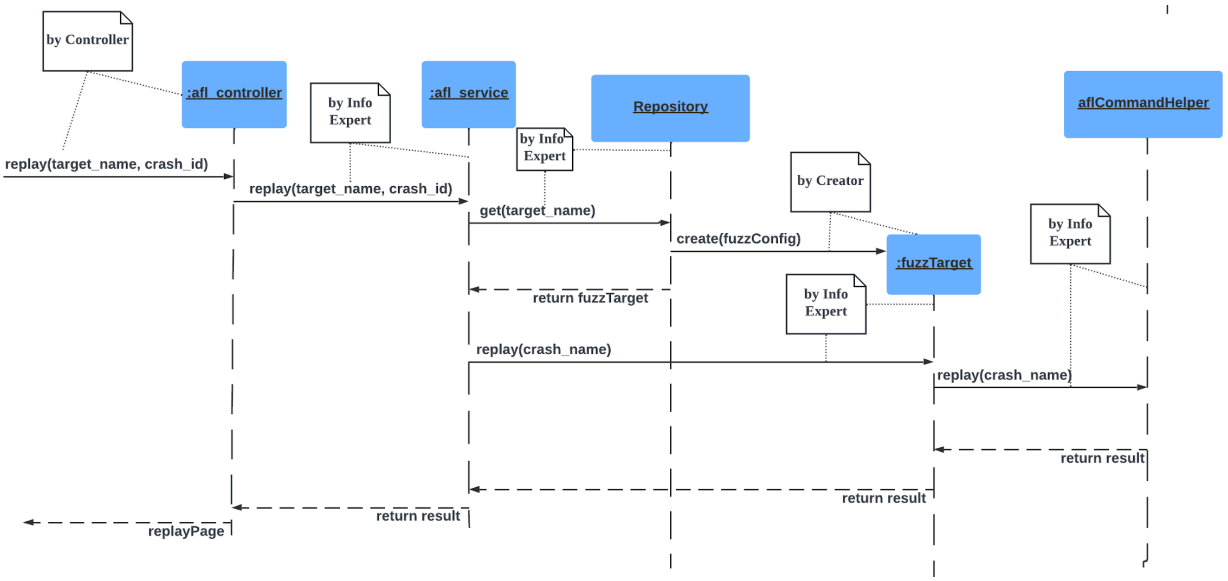
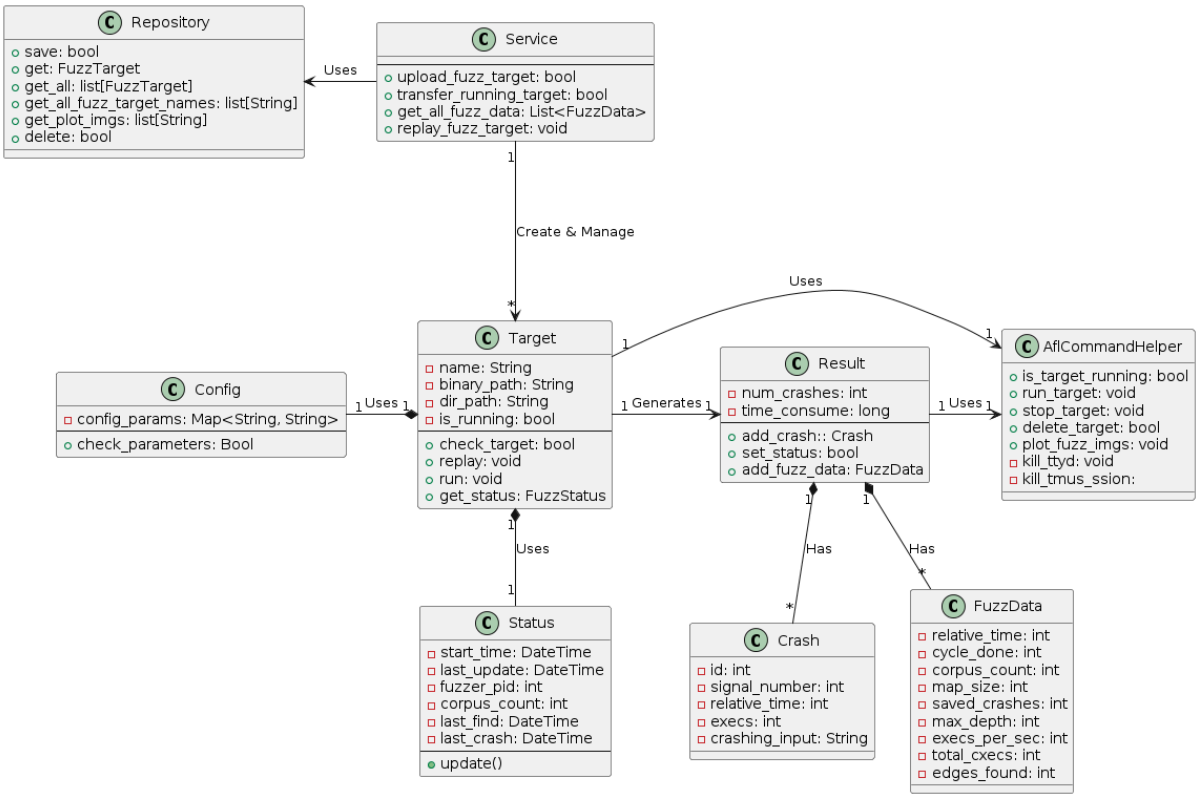


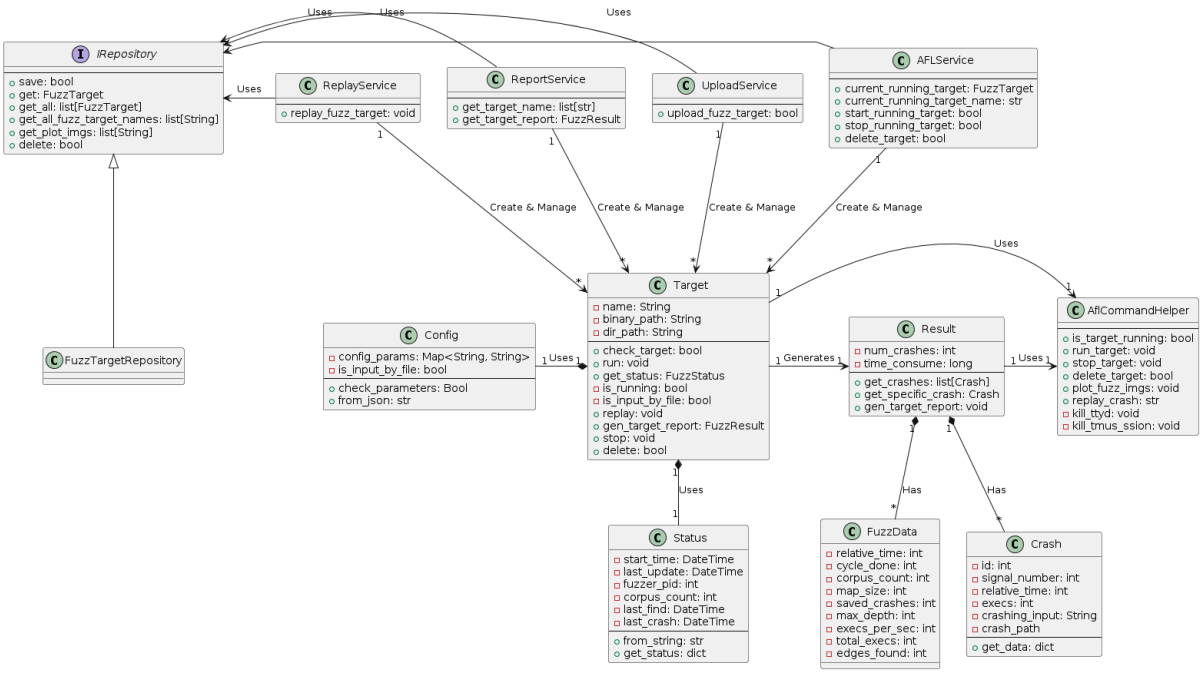
Figure 3.2.2-5. replay

3.3. Design class model



4. Implementation Class Model

4.1. Implementation class model



4.2. difference between implementation class model and design class model

4.2.1. Comparison with design and implementation class

Class	Method	Design	Impl.
Sevice	upload_fuzz_target	Y	N
	transfer_running_target	Y	N
	get_all_fuzz_data	Y	N
	replay_fuzz_tartget	Y	N
AFLService	current_running_target	N	Y
	current_running_target_name	N	Y
	transfer_running_target	N	Y
	delete_target	N	Y
	start_running_target	N	Y
UploadService	upload_fuzz_target	N	Y
ReportServie	get_target_name	N	Y
	get_target_report	N	Y
ReplayService	replay_fuzz_target	N	Y
Target	check_target_bin	Y	N
	replay	Y	Y
	run	Y	Y
	get_target_status	Y	Y
	delete	N	Y
	gen_target_report	N	Y
	is_running	N	Y
	is_input_by_file	N	Y
	stop	N	Y

Config	check_parameters	Y	Y
	from_json	N	Y
Status	update	Y	N
	from_string	N	Y
	get_status	N	Y
Result	add_crash	Y	N
	set_status	Y	N
	add_fuzz_data	Y	N
	_plot_img	N	Y
	gen_target_report	N	Y
	get_crashes	N	Y
	gen_target_report	N	Y
Crash	get_data	N	Y
IRepository	save	Y	Y
	get	Y	Y
	get_all	Y	Y
	get_all_fuzz_target_names	Y	Y
	get_plot_imgs	Y	Y
	delete	Y	Y
afl_command_helper	is_target_running	Y	Y
	run_target	Y	Y
	stop_target	Y	Y
	delete_target	Y	Y
	plot_fuzz_imgs	Y	Y
	kill_ttyd	Y	Y
	kill_tmux_session	Y	Y

	replay_crash	N	Y

4.2.2. Summary of implementation class/method changed

	Number of added	Number of removed	Number of modified
Class	5	1	0
Method	24	9	0

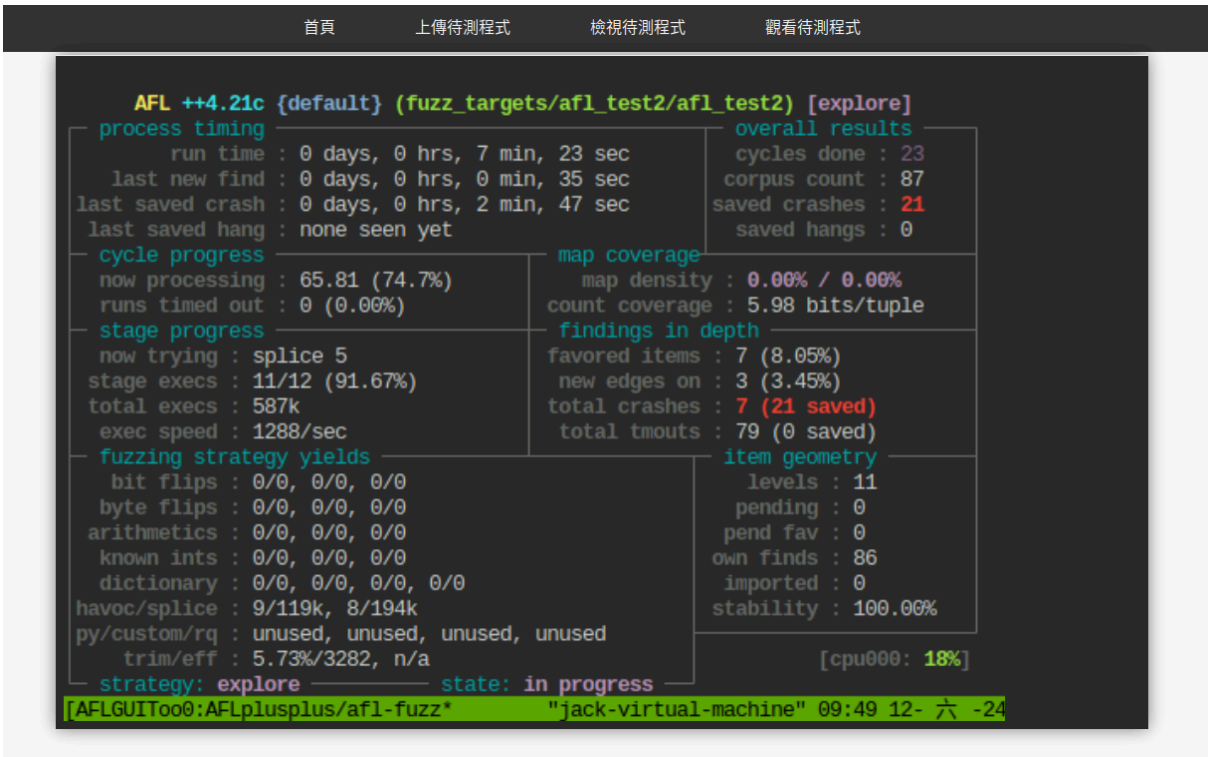
4.3. Calculate Line of Code

No	Class Name	Number of methods	Line of Code in Class (without comment)
1	AFLservice	5	52
2	target	9	55
3	config	2	12
4	status	2	128
5	result	4	34
6	fuzzData	1	12
7	crash	2	19
8	IRepository	7	107
9	FuzzTargetRepository	6	30
10	afl_command_helper	14	167
11	UploadService	2	16
12	ReportService	3	24
13	ReplayService	2	12

5. Programming

5.1. Snapshots of system execution

首頁
上傳待測程式
檢視待測程式
觀看待測程式
Reports Page
刪除所選目標
選擇
名稱
正在執行
查看詳細狀態
執行
中止
afl_test2
是
查看詳細狀態
執行
中止
afl_test1
查看詳細狀態
執行

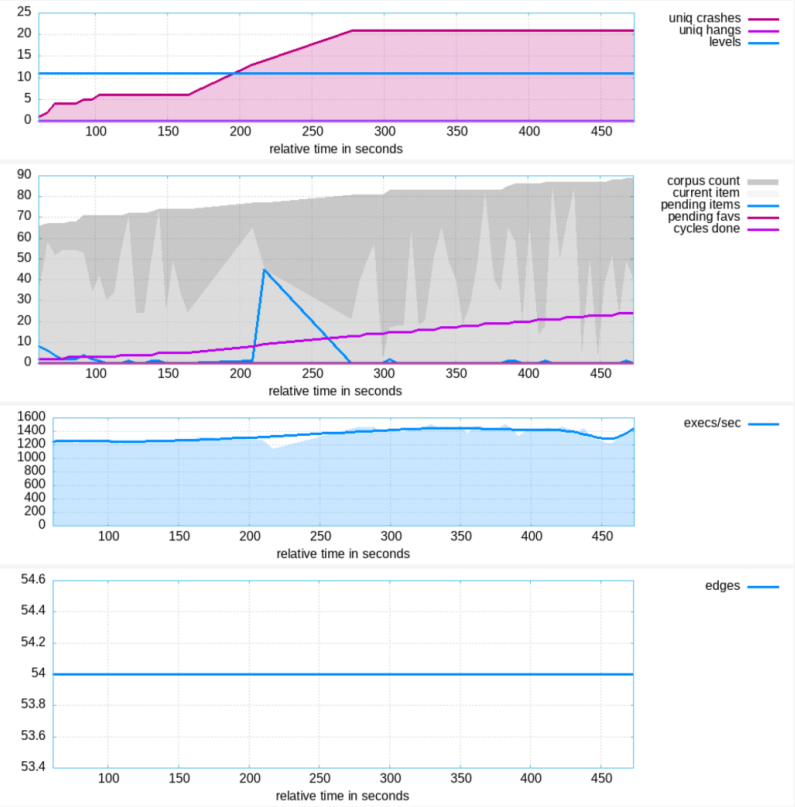

首頁
上傳待測程式
檢視待測程式
觀看待測程式
AFL ++4.21c {default} (fuzz_targets/afl_test2/afl_test2) [explore]
process timing
run time : 0 days, 0 hrs, 7 min, 23 sec
last new find : 0 days, 0 hrs, 0 min, 35 sec
last saved crash : 0 days, 0 hrs, 2 min, 47 sec
last saved hang : none seen yet
overall results
cycles done : 23
corpus count : 87
saved crashes : 21
saved hangs : 0
cycle progress
now processing : 65.81 (74.7%)
runs timed out : 0 (0.00%)
map coverage
map density : 0.00% / 0.00%
count coverage : 5.98 bits/tuple
stage progress
now trying : splice 5
stage execs : 11/12 (91.67%)
total execs : 587k
exec speed : 1288/sec
findings in depth
favored items : 7 (8.05%)
new edges on : 3 (3.45%)
total crashes : 7 (21 saved)
total tmouts : 79 (0 saved)
fuzzing strategy yields
bit flips : 0/0, 0/0, 0/0
byte flips : 0/0, 0/0, 0/0
arithmetics : 0/0, 0/0, 0/0
known ints : 0/0, 0/0, 0/0
dictionary : 0/0, 0/0, 0/0, 0/0
havoc/splice : 9/119k, 8/194k
py/custom/rq : unused, unused, unused, unused
trim/eff : 5.73%/3282, n/a
item geometry
levels : 11
pending : 0
pend fav : 0
own finds : 86
imported : 0
stability : 100.00%
[cpu000: 18%]
strategy: explore
state: in progress
[AFLGUIToo0:AFLplusplus/afl-fuzz* "jack-virtual-machine" 09:49 12- 六 -24

首頁 上傳待測程式 檢視待測程式 觀看待測程式

Report Page

Target Name: afl_test2

Fuzz Result:



Number of crashed found: 21

Crashed Targets List:

Replay

Crash ID: 20

Signal Number: 6

Relative Time: 275585

Execs: 348121

Crashing Input: cracrr cras cr cras cras' crashas cras K crercrahrr cf cra cras $ cras cras cwcrK crasK cras cfa cras craasK cras cfaaaaaaaaaxaaaaaaaaaaaaaaa cr"""""""""""""""""""""""a

Replay

Crash ID: 19

Signal Number: 6

Relative Time: 250719

Execs: 313526

Crashing Input: crarrKk)cra crcrr cra cra cras K crercrahrr c{ cras Ka cr cras cf crashrr c{ cras Ka cra cra cfa cra

首頁 上傳待測程式 檢視待測程式 觀看待測程式

Replay target afl_test2 for crash Id 20

If not showing the crash replay result, you can refresh the page again.

```
Script started on 2024-06-12 09:52:36+08:00 [TERM="xterm-256color" TTY="/dev/pts/5" COLUMNS="-1" LINES="-1"]
                                                        rrcarc
                                                         sarc
                                                          rc
                                                         sarc
                                                        'sarc
bash: line 1: 582567 Done                    cat fuzz_targets/afl_test2/output/default/crashes/id:000020,sig:06,src:000021+000005,time:275585,execs:348121,op:splice,rep:8
                                                582569 Aborted                 (core dumped) | fuzz_targets/afl_test2/afl_test2

                                                     Script done on 2024-06-12 09:52:36+08:00 [COMMAND_EXIT_CODE="134"]
```

5.2. Source Code Listing (note: select and list the most important source code;no more than 10 pages)

- afl_controller.py

```python
@bp.route('/')
def index():
    return render_template('index.html')


@bp.route('/upload', methods=['GET', 'POST'])
def upload():
    if request.method == 'GET':
        return render_template('upload.html')
    file = request.files['file']
    is_input_by_file = request.form.get('isInputByFile') == 'on'
    if file:
        result = uploadService.upload_fuzz_Target(file, is_input_by_file)
        if result:
            flash(f'{file.filename} 上傳成功')
        else:
            flash(f'{file.filename} 上傳失敗')
    return redirect(url_for('afl_controller.upload'))


@bp.route('/observe', methods=['GET', 'POST'])
def observe():
    return render_template('observe.html')


@bp.route('/reports', methods=['GET'])
def reports():
    target_names = reportService.get_target_names()
    current_target_name = aflService.current_running_target_name()
    return render_template('reports.html', targetNames=target_names, currentTargetName=current_target_name)


@bp.route('/reports/<targetName>', methods=['GET'])
def target_report(targetName):
    target_report = reportService.get_target_report(targetName)
    target_report_img_path = reportService.get_target_report_img_path(
        targetName)
    return render_template('target_report.html', target=targetName, targetResult=target_report, imgs=target_report_img_path)


@bp.route('/replay', methods=['GET'])
def replay():
    target_name = request.args.get('targetName')
    crash_id = request.args.get('crashId')
    target_report_content = replayService.replay_target(
        target_name, crash_id)
    return render_template('replay.html', targetName=target_name, crash=crash_id, htmlContent=target_report_content)


@bp.route('/execute', methods=['POST'])
def execute():
    target_name = request.form.get('targetName')
    aflService.start_running_target(target_name)
    return redirect(url_for('afl_controller.observe'))


@bp.route('/delete', methods=['POST'])
def delete():
    delete_target_list = request.get_json()
    aflService.delete_target(delete_target_list)
    return redirect(url_for('afl_controller.reports'))


@bp.route('/stop', methods=['POST'])
def stop():
    aflService.stop_running_target()
    return redirect(url_for('afl_controller.reports'))


fuzzTargetRepository = FuzzTargetRepository()
aflService = AFLService(fuzzTargetRepository)
uploadService = UploadService(fuzzTargetRepository)
reportService = ReportService(fuzzTargetRepository)
replayService = ReplayService(fuzzTargetRepository)
```

- afl_service.py

```python
class AFLService:
    def __init__(self, repository: IRepository):
        self.repository = repository

    def current_running_target(self) -> FuzzTarget:
        targets = self.repository.get_all()
        for target in targets:
            if target.is_running():
                return target
        return None

    def current_running_target_name(self) -> str:
        target = self.current_running_target()
        if target is None:
            return None
        return target.name

    def start_running_target(self, target_name: str) -> bool:
        try:
            target = self.repository.get(target_name)
            if target is None:
                return False
            return target.run()
        except Exception as e:
            print(f"Error transferring running target: {e}")
            return False

    def stop_running_target(self) -> bool:
        target = self.current_running_target()
        if target is None:
            return False
        return target.stop()

    def delete_target(self, target_name: list[str]) -> bool:
        try:
            for name in target_name:
                target = self.repository.get(name)
                if target is None:
                    return False
                target.delete()
            return True
        except Exception as e:
            print(f"Error deleting target: {e}")
            return False
```

- fuzzTarget.py

```python
class FuzzTarget:
    def __init__(self, name: str, dirPath: Path, fuzzResult: FuzzResult, fuzzStatus: FuzzStatus, fuzzConfig: FuzzConfig = None):
        self.name: str = name
        self.dir_path: Path = dirPath
        self.binary_path: Path = dirPath/name
        self.fuzz_result: FuzzResult = fuzzResult
        self.fuzz_status: FuzzStatus = fuzzStatus
        self.fuzz_config: FuzzConfig = fuzzConfig
        self.process = None
        self.__is_running: bool = None

    def run(self) -> bool:
        return afl_command_helper.run_target(self)

    def is_running(self) -> bool:
        self.__is_running = afl_command_helper.is_target_running(self.name)
        return self.__is_running

    def is_input_by_file(self) -> bool:
        return self.fuzz_config.is_input_by_file

    def stop(self) -> None:
        afl_command_helper.stop_target()
        return

    def gen_target_report(self) -> FuzzResult:
        if (self.fuzz_result is None):
            return None
        self.fuzz_result.gen_target_report(self.name)
        return self.fuzz_result

    def delete(self) -> bool:
        return afl_command_helper.delete_target(self.name)

    def replay(self, crash_id: int) -> str:
        crash = self.fuzz_result.get_specific_crash(id=crash_id)
        replay_content = afl_command_helper.replay_crash(
            self.name, crash.crash_path, self.is_input_by_file())
        return replay_content
```

- fuzzResult.py

```python
class FuzzResult:
    def __init__(self, crashes: list[Crash] = [], fuzzData: list[FuzzData] = []):
        self.__crashes: list[Crash] = crashes
        self.__fuzzData: list[FuzzData] = fuzzData

    @property
    def num_crashes(self) -> int:
        return len(self.__crashes)

    @property
    def get_crashed(self) -> list[Crash]:
        return self.__crashes

    @property
    def num_fuzz_data(self) -> int:
        return len(self.__fuzzData)

    def get_specific_crash(self, id: int) -> Crash:
        for crash in self.__crashes:
            if crash.id == int(id):
                return crash
        return None

    def _plot_imgs(self, target_name: str) -> None:
        afl_command_helper.plot_fuzz_imgs(target_name)

    def gen_target_report(self, target_name: str):
        self._plot_imgs(target_name)
```

- IRepository.py

```python
class IRepository(ABC):
    @abstractmethod
    def save(self, file: FileStorage, path: Path, config_json: str) -> bool:
        raise NotImplementedError

    @abstractmethod
    def get(self, target_name: str) -> FuzzTarget:
        raise NotImplementedError

    @abstractmethod
    def get_all(self) -> list[FuzzTarget]:
        raise NotImplementedError

    @abstractmethod
    def get_all_fuzz_target_names(self) -> list[str]:
        raise NotImplementedError

    @abstractmethod
    def get_plot_imgs(self, target_name: str) -> list[str]:
        raise NotImplementedError

    @abstractmethod
    def delete(self, target_name: str) -> bool:
        raise NotImplementedError
```

- fuzz_target_repository.py

```python
class FuzzTargetRepository(IRepository):
    def save(self, file: FileStorage, path: Path, config_json: str) -> bool:
        if file is None or file.filename is None or file.filename == '' or path.exists():
            return False
        try:
            path.mkdir(parents=True, exist_ok=True)
            save_path = path / file.filename
            file.save(save_path)
            save_path.chmod(0o777)
            self.__set_default_dir_and_seed(path)
            self.__set_config_json(path, config_json)
            return True
        except Exception as e:
            return False

    def get(self, target_name: str) -> FuzzTarget | None:
        target_dir = Path(current_app.config['UPLOAD_FOLDER'])
        target_path = target_dir / target_name
        if not target_path.exists():
            return None

        fuzz_result = self.__get_fuzz_result_by_name(target_name)
        fuzz_status = self.__get_fuzz_status_by_name(target_name)
        fuzz_config = self.__get_fuzz_config_by_name(target_name)
        return FuzzTarget(
            target_name, target_path, fuzz_result, fuzz_status, fuzz_config)

    def get_all(self) -> list[FuzzTarget]:
        names = self.get_all_fuzz_target_names()
        targets = []
        for name in names:
            target = self.get(name)
            if target:
                targets.append(target)
        return targets

    def get_all_fuzz_target_names(self) -> list[str]:
        target_dir = Path(current_app.config['UPLOAD_FOLDER'])
        return [str(dir.name) for dir in target_dir.iterdir() if dir.is_dir()]

    def get_plot_imgs(self, target_name: str) -> list[str]:
        target_img_dir = Path(
            current_app.config['TARGET_IMG_FOLDER']) / target_name
        if not target_img_dir.exists():
            return []
        return [img.name for img in target_img_dir.iterdir() if img.name.endswith(('.png', '.jpg'))]

    def delete(self, target_name: str) -> bool:
        target = Path(current_app.config['UPLOAD_FOLDER']) / target_name
        if target.exists():
            try:
                shutil.rmtree(target)
                return True
            except Exception as e:
                return False
        return False
```

6. Unit Testing

6.1. Snapshots of testing result

Coverage report: 83%

Files | Functions | Classes

coverage.py v7.5.3, created at 2024-06-12 16:41 +0800

File	statements	missing	excluded	coverage ▼
app/controllers/__init__.py	0	0	0	100%
app/models/__init__.py	0	0	0	100%
app/services/__init__.py	0	0	0	100%
tests/__init__.py	0	0	0	100%
tests/test_crash.py	5	0	0	100%
app/models/fuzzConfig.py	8	0	0	100%
app/services/replay_service.py	8	0	0	100%
tests/test_fuzzConfig.py	9	0	0	100%
app/models/crash.py	11	0	0	100%
app/models/fuzzData.py	11	0	0	100%
app/services/upload_service.py	12	0	0	100%
tests/test_fuzzStatus.py	13	0	0	100%
tests/test_replay_service.py	14	0	0	100%
app/services/report_service.py	18	0	0	100%
tests/test_upload_service.py	20	0	0	100%
app/models/fuzzResult.py	25	0	0	100%
tests/test_fuzzResult.py	26	0	0	100%
app/models/fuzzTarget.py	36	0	0	100%
tests/test_report_service.py	41	0	0	100%
tests/test_afl_command_helper.py	48	0	0	100%
tests/test_fuzz_target_repository.py	51	0	0	100%
tests/test_fuzzTarget.py	54	0	0	100%
app/models/fuzzStatus.py	64	0	0	100%
tests/test_afl_service.py	96	0	0	100%
app/services/afl_service.py	42	3	0	93%
app/Repository/IRepository.py	23	6	0	74%
app/helper/afl_command_helper.py	105	49	0	53%
app/controllers/afl_controller.py	60	30	0	50%
app/Repository/fuzz_target_repository.py	117	65	0	44%
app/__init__.py	15	9	0	40%
Total	**932**	**162**	**0**	**83%**

6.2. Unit Testing Code Listing

- test_crash.py

```python
def test_crash_get_data():
    crash = Crash(id=1, signal_number=11, relative_time=100,
                  execs=5, crashingInput="input", crash_path="path")
    expected_data = {
        "id": 1,
        "signal_number": 11,
        "relative_time": 100,
        "execs": 5,
        "crashingInput": "input"
    }
    assert crash.get_data == expected_data
```

- test_fuzzResult.py

```python
@pytest.fixture
def fuzz_result():
    crashes = [
        Crash(id=2, signal_number=11, relative_time=100,
              execs=5, crashingInput="input", crash_path="path"),
        Crash(id=3, signal_number=11, relative_time=100,
              execs=5, crashingInput="input", crash_path="path")
    ]
    fuzz_data = [
        FuzzData(1, 11, 100, 5, 0.1, 0.2, 0.3, 0.4, 0.5),
        FuzzData(2, 22, 200, 10, 0.2, 0.3, 0.4, 0.5, 0.6)
    ]
    return FuzzResult(crashes, fuzz_data)


def test_num_crashes(fuzz_result):
    assert fuzz_result.num_crashes == 2


def test_get_crashed(fuzz_result):
    assert len(fuzz_result.get_crashed) == 2


def test_num_fuzz_data(fuzz_result):
    assert fuzz_result.num_fuzz_data == 2


def test_gen_target_report(fuzz_result):
    # Arrange
    with patch("app.models.fuzzResult.afl_command_helper.plot_fuzz_imgs") as mock:

        # Act
        fuzz_result.gen_target_report("test")

        # Assert
        mock.assert_called_once_with("test")


def test_get_specific_crash(fuzz_result):
    # Arrange
    id = 2

    # Act
    result = fuzz_result.get_specific_crash(id)

    # Assert
    assert result.id == id


def test_get_specific_crash_with_bad_id(fuzz_result):
    # Arrange
    id = 5

    # Act
    result = fuzz_result.get_specific_crash(id)

    # Assert
    assert result is None
```

- test_fuzzConfig.py

```python
def test_fuzzConfig():
    # Arrange
    fuzz_config = FuzzConfig(is_input_by_file=True)
    # Act
    result = fuzz_config.is_input_by_file
    # Assert
    assert result == True


def test_fuzzConfig_from_json():
    # Arrange
    json_str = '{"isInputByFile": true}'
    # Act
    result = FuzzConfig.from_json(json_str)
    # Assert
    assert result.is_input_by_file == True
```

- test_replay_service.py

```python
@pytest.fixture
def repository():
    return Mock(spec=IRepository)


def test_replay_target(repository):
    # Arrange
    replay_service = ReplayService(repository)
    target_name = "test"
    crash_id = 1

    # Act
    replay_service.replay_target(target_name, crash_id)

    # Assert
    repository.get.assert_called_once_with(target_name)
    repository.get.return_value.replay.assert_called_once_with(crash_id)
```

- test_upload_service.py

```python
@pytest.fixture
def repository():
    return Mock(spec=IRepository)


def test_upload_fuzz_Target(repository):
    # Arrange
    app = Flask('test')
    with app.app_context():
        with patch("app.services.upload_service.current_app.config", {"UPLOAD_FOLDER": "uploads"}):
            file = FileStorage(filename='test.txt')
            repository.save.return_value = True
            upload_service = UploadService(repository=repository)

            # Act
            result = upload_service.upload_fuzz_Target(
                file, is_input_by_file=True)

            # Assert
            assert result == True
            repository.save.assert_called_once_with(
                file, Path("uploads/test.txt"), '{\n    "isInputByFile": true\n}')
```

- test_fuzzTarget.py (Part 1)

```python
@pytest.fixture
def fuzz_target():
    fuzz_config = FuzzConfig(is_input_by_file=True)
    return FuzzTarget(name="test", dirPath=Path(""), fuzzResult=None, fuzzStatus=None, fuzzConfig=fuzz_config)


def test_run(fuzz_target):
    # Arrange
    with patch("app.models.fuzzTarget.afl_command_helper.run_target") as mock:
        mock.return_value = True

        # Act
        result = fuzz_target.run()

        # Assert
        assert result == True
        mock.assert_called_once_with(fuzz_target)


def test_is_running(fuzz_target):
    # Arrange
    with patch("app.models.fuzzTarget.afl_command_helper.is_target_running") as mock:
        mock.return_value = True

        # Act
        result = fuzz_target.is_running()

        # Assert
        assert result == True
        mock.assert_called_once_with("test")


def test_stop(fuzz_target):
    # Arrange
    with patch("app.models.fuzzTarget.afl_command_helper.stop_target") as mock:

        # Act
        fuzz_target.stop()

        # Assert
        mock.assert_called_once()d=1)
        mock.assert_called_once_with("test", "fake_path", True)
```

- test_fuzzTarget.py (Part 2)

```python
def test_gen_target_report(fuzz_target):
    # Arrange
    fake_fuzz_result = Mock(spec=FuzzResult)
    fuzz_target.fuzz_result = fake_fuzz_result

    # Act
    fuzz_target.gen_target_report()

    # Assert
    fake_fuzz_result.gen_target_report.assert_called_once_with("test")


def test_gen_target_report_none(fuzz_target):
    # Act
    result = fuzz_target.gen_target_report()

    # Assert
    assert result == None


def test_delete(fuzz_target):
    # Arrange
    with patch("app.models.fuzzTarget.afl_command_helper.delete_target") as mock:
        mock.return_value = True

        # Act
        result = fuzz_target.delete()

        # Assert
        assert result == True
        mock.assert_called_once_with("test")


def test_is_input_by_file(fuzz_target):
    # Act
    result = fuzz_target.is_input_by_file()

    # Assert
    assert result == True


def test_replay(fuzz_target):
    # Arrange
    with patch("app.models.fuzzTarget.afl_command_helper.replay_crash") as mock:
        fake_fuzz_result = Mock(spec=FuzzResult)
        fake_fuzz_result.get_specific_crash.return_value = Crash(id=1, signal_number=11, relative_time=100,
                                                                 execs=5, crashingInput="input", crash_path="fake_path")
        fuzz_target.fuzz_result = fake_fuzz_result
        mock.return_value = "fake_replay_content"

        # Act
        result = fuzz_target.replay(1)

        # Assert
        assert result == "fake_replay_content"
        fake_fuzz_result.get_specific_crash.assert_called_once_with(id=1)
        mock.assert_called_once_with("test", "fake_path", True)
```

- Measurement

112598052 林祐清	**112598010** 林建澄
HW 1	
2024/02/24 14:00~16:00	2024/02/24 14:00~16:00
2 H	2 H
HW 2	
2024/03/09 14:00~16:00	2024/03/09 14:00~16:00
2024/03/10 14:00~16:00	2024/03/10 14:00~16:00
2024/03/12 13:00~16:00	2024/03/12 13:00~16:00
7 H	7 H
HW 3	
2024/03/23 13:00~18:00	2024/03/23 13:00~18:00
5 H	5 H
HW 4	
2024/04/13 13:00~18:00	2024/04/13 13:00~18:00
2024/04/14 13:00~18:00	2024/04/14 13:00~18:00
2024/04/16 13:00~18:00	2024/04/16 13:00~18:00
15 H	15 H
HW 5	
2024/04/20 10:00~18:00	2024/04/20 10:00~18:00
2024/04/21 17:00~21:00	2024/04/21 17:00~21:00
2024/04/24 19:00~22:30	2024/04/24 19:00~22:30
2024/04/26 13:30~17:30	2024/04/26 13:30~17:30
2024/04/27 20:00~00:00	2024/04/27 20:00~00:00
2024/04/28 11:00~21:30	2024/04/28 11:00~21:30

32 H	32 H
HW 6	
2024/05/18 10:00~18:00	2024/05/18 10:00~18:00
2024/05/19 10:00~18:00	2024/05/19 10:00~18:00
16 H	16 H
HW 7	
2024/06/06 18:00~23:00	2024/06/06 18:00~23:00
2024/06/07 18:00~23:00	2024/06/07 18:00~23:00
5 H	5 H
Total Time Spent	
82 H	82 H