
System Model (Sequence Diagram) Document

Project Name	Fuzz Testing을 통한 위성 SW 분석
-----------------	---------------------------

05 조

202002473 김승혁

201902733 이정윤

202002699 조민기

지도교수: 이성호 교수님 (서명)



Document Revision History

REV#	DATE	AFFECTED SECTION	AUTHOR
1	2025/04/29	초안 작성	김승혁
2	2025/04/30	연구 알고리즘 순서도	김승혁

Table of Contents

1.	INTRODUCTION.....	5
1.1.	연구 배경	5
1.2.	연구 목적	5
1.3.	연구 질문/ 가설	6
2.	USE CASE DIAGRAM	7
2.1.	소프트웨어 활용 사례	7
2.2.	문제 해결에 대한 사용 사례 DIAGRAM.....	7
3.	SEQUENCE DIAGRAM	8
3.1.	해결 방법에 대한 알고리즘 순서도	8

List of Figure

그림 1 소프트웨어 사용 사례 DIAGRAM.....	7
그림 2. 문제 해결에 대한 사용 사례 DIAGRAM.....	7

1. Introduction

1.1. 연구 배경

최근 저비용 위성 발사와 소형 위성 기술의 발전으로 우주 산업의 문턱이 낮아지면서, NASA의 fprime과 같은 오픈소스 위성 비행 소프트웨어의 활용이 증가하고 있다. fprime은 컴포넌트 기반 아키텍처와 직접 통신(fpp) 방식을 특징으로 하며, 소형 위성 개발에 적합하지만, 검증 사례나 특화된 테스트 도구가 부족하다. 특히, 퍼징(Fuzzing)은 예기치 않은 입력값으로 소프트웨어의 잠재적 결함을 발견하는 효과적인 테스트 기법이지만, fprime의 고유 특성과 컴포넌트 기반 설계를 고려한 퍼징 연구는 초기 단계 수준이다. 따라서 위성 비행 소프트웨어에 특화된 효과적인 퍼징 전략을 연구하는 것이 중요한 과제가 된다.

1.2. 연구 목적

본 연구의 목적은 fprime 위성 비행 소프트웨어 환경에서 다양한 퍼징 도구들의 성능을 비교 분석하고, fprime의 특성을 고려한 효과적인 퍼징 적용 전략을 제시하는 것입니다. 특히, 위성 SW 개발자의 결함 탐지 문제를 해결하거나, 퍼징 기법의 효과성을 검증함으로써 실용적 기여를 목표로 합니다. 이를 통해 위성 SW의 신뢰성 검증을 강화하고, 초기 단계에서 잠재적 결함을 발견할 가능성을 높이하고자 합니다.

1.3. 연구 질문/ 가설

본 연구는 다음과 같은 연구 질문에 답하고자 한다:

- **RQ1.**
fprime 위성 비행 소프트웨어 환경에서 다양한 퍼징 도구를 활용한 결함 탐지 활동이 기존의 테스트 방식에 비해 결함 탐지율에 어떠한 영향을 미치는가?
- **RQ2.**
퍼징 도구의 유형(예: Coverage-Guided, Generation-Based)이 fprime의 컴포넌트 기반 아키텍처와 통신 방식(fpp)에 따라 결함 탐지 성능에서 어떤 구체적인 차이를 보일 것인가?

본 연구는 다음과 같은 가설을 설정할 수 있다:

- **H1.**
fprime 환경에서 제안하는 퍼징 전략을 활용한 결함 탐지 활동이 기존의 테스트 방식보다 결함 탐지율을 유의미하게 향상시킬 것이다.
- **H2.**
Generation-Based 퍼징 도구는 fprime의 컴포넌트 간 통신 방식(fpp)을 고려할 때, 더 높은 상태 탐색 능력과 결함 탐지 성능을 보일 것이다.

2. Use Case Diagram

2.1. 소프트웨어 활용 사례

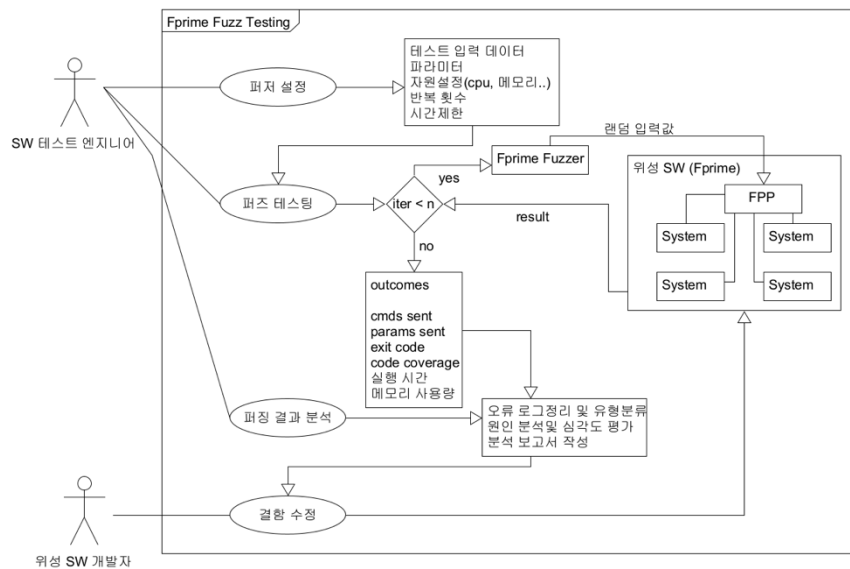


그림 1 소프트웨어 사용 사례 Diagram

2.2. 문제 해결에 대한 사용 사례 Diagram

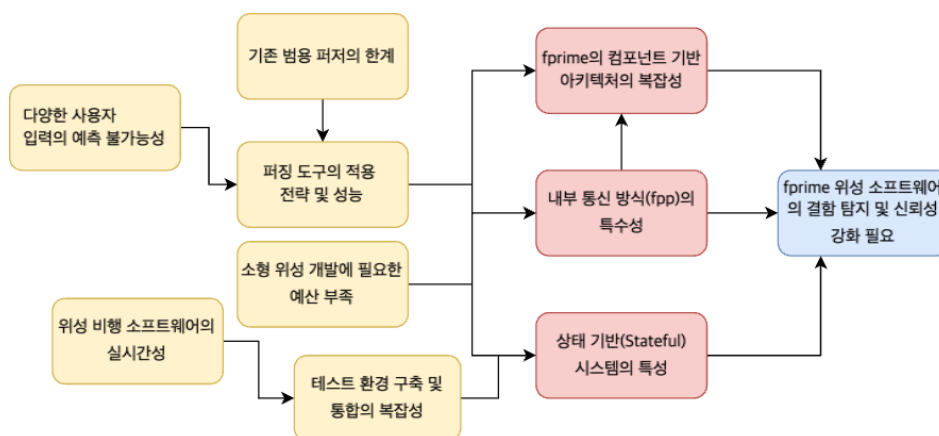
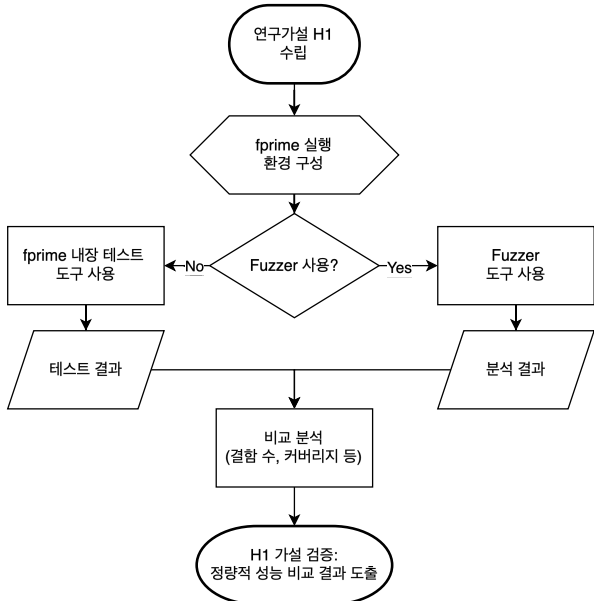


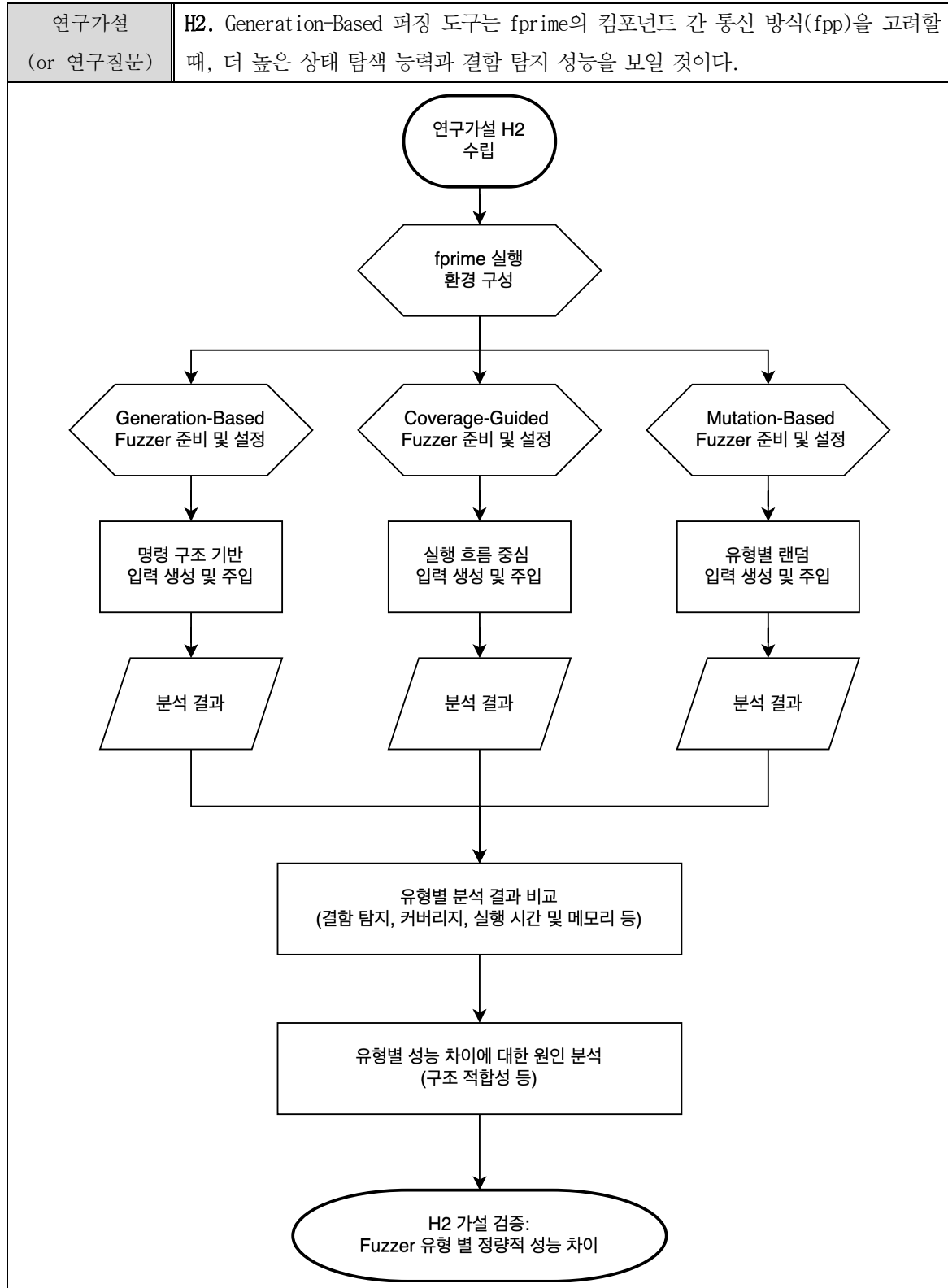
그림 2. 문제 해결에 대한 사용 사례 Diagram

3. Sequence Diagram

3.1. 해결 방법에 대한 알고리즘 순서도(H1)

연구가설 (or 연구질문)	H1. fprime 환경에서 제안하는 퍼징 전략을 활용한 결함 탐지 활동이 기존의 테스트 방식보다 결함 탐지율을 유의미하게 향상시킬 것이다.
	 <pre> graph TD A([연구가설 H1 수립]) --> B{{fprime 실행 환경 구성}} B --> C{Fuzzer 사용?} C -- Yes --> D[Fuzzer 도구 사용] C -- No --> E[fprime 내장 테스트 도구 사용] D --> F[/분석 결과/] E --> G[/테스트 결과/] F --> H[비교 분석
(결함 수, 커버리지 등)] G --> H H --> I([H1 가설 검증:
정량적 성능 비교 결과 도출]) </pre>
핵심 문제 정의	1. 퍼저를 사용함으로써 결함 탐지율에 어떤 정량적 차이가 발생하는지 평가
알고리즘 순서	1. SW 테스트 엔지니어: 연구가설(H1) 수립 Fprime 실행 환경 구성 (A) fprime 기본 테스트 케이스 설계 및 실행 (B) 퍼저 설정 및 자동 테스트 실행 2. 위성 SW(Fprime): 입력값 처리 및 결과 반환 3. SW 테스트 엔지니어: 결과분석(결함 탐지, 커버리지, 자원 사용량 등) 두 방식의 테스트 경험 및 결과 비교

3.2. 해결 방법에 대한 알고리즘 순서도(H2)



핵심 문제 정의	<p>1. 퍼저 유형별로 fprime의 컴포넌트 아키텍처와 통신 방식(fpp)에 따라 결함 탐지율, 코드 커버리지, 자원 사용량(실행시간, 메모리 등)에 어떤 정량적 차이가 발생하는지 평가</p> <p>2. 어떤 퍼저 유형이 fprime 구조에 더 효과적인지 실험적으로 검증</p>
알고리즘 순서	<p>1. SW 테스트 엔지니어:</p> <p>연구 가설(H2) 수립</p> <p>Fprime 실행 환경 구성</p> <p>2. 퍼저(Fuzzer):</p> <p>유형별 퍼저 준비 및 설정(입력 생성 전략, 반복 횟수, 자원 제한 등)</p> <p>유형별 입력값 생성 및 fprime 시스템에 주입</p> <p>3. 위성 SW(Fprime):</p> <p>입력값 처리, 분석 결과 반환</p> <p>4. SW 테스트 엔지니어:</p> <p>분석 결과 수집(탐지된 결함, 커버리지, 실행시간, 메모리 등)</p> <p>유형별 분석 결과 비교(결함 탐지율, 커버리지, 자원 사용량 등 정량적 평가)</p> <p>유형별 성능 차이에 대한 원인 분석(구조 적합성 등)</p> <p>가설 검증 및 보고서 작성</p>