KCSE 2012
2012.2.8~10

소프트웨어 역공학을 통한 HeliScope OFP 테스트 케이스 생성

이종훈, 이동아, 유준범, 송승화, 김두현
College of Information and Communication
KONKUK UNIVERSITY

Dependable Software Laboratory

차례

- 서론
- 배경 연구
  - HeliScope Project
  - HeliScope Project 비행 운용 프로그램을 위한 검증 절차
  - 소프트웨어 역공학
  - 소프트웨어 테스팅
- 적용 및 결과
  - 소프트웨어 역공학 적용
  - 소프트웨어 테스팅 적용
- 결론 및 향후 연구

소프트웨어 역공학을 통한 HeliScope OFP 테스트 케이스 생성

서론

서론

- 소프트웨어 동작의 정확성 검증
  - 다양한 분야에서 소프트웨어 사용 비중 증가
  - 소프트웨어의 결함이 사고 발생으로 연결
    - 인명 및 경제적 피해 발생 등

- HeliScope OFP(Operational Flight Program)
  - 무인 헬리콥터의 비행 제어를 위한 소프트웨어
  - OFP의 결함이 비행 제어의 이상으로 연결될 수 있음
  - 비행체의 오동작은 임무 실패나 추락 등의 사고로 이어질 수 있음
    - 소프트웨어 동작 정확성 검증이 필요

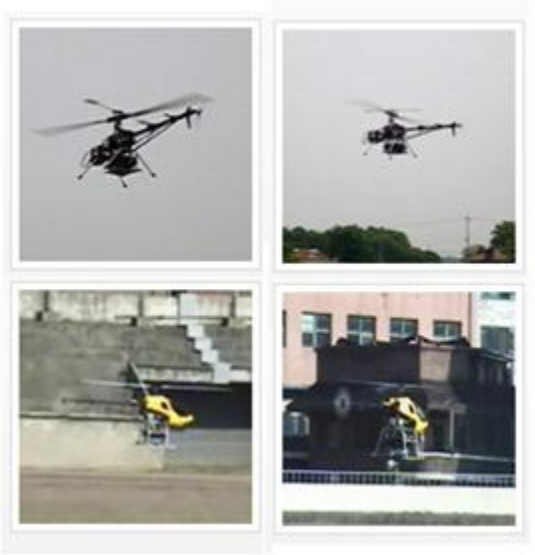
소프트웨어 역공학을 통한 HeliScope OFP 테스트 케이스 생성

배경 연구

배경 연구

- HeliScope Project
  - 방재, 재난 복구 등의 임무 수행을 위한 소형 무인 헬리콥터 개발 프로젝트
  - 직접 조종 이외에 자동 항법(무인 조종) 기능

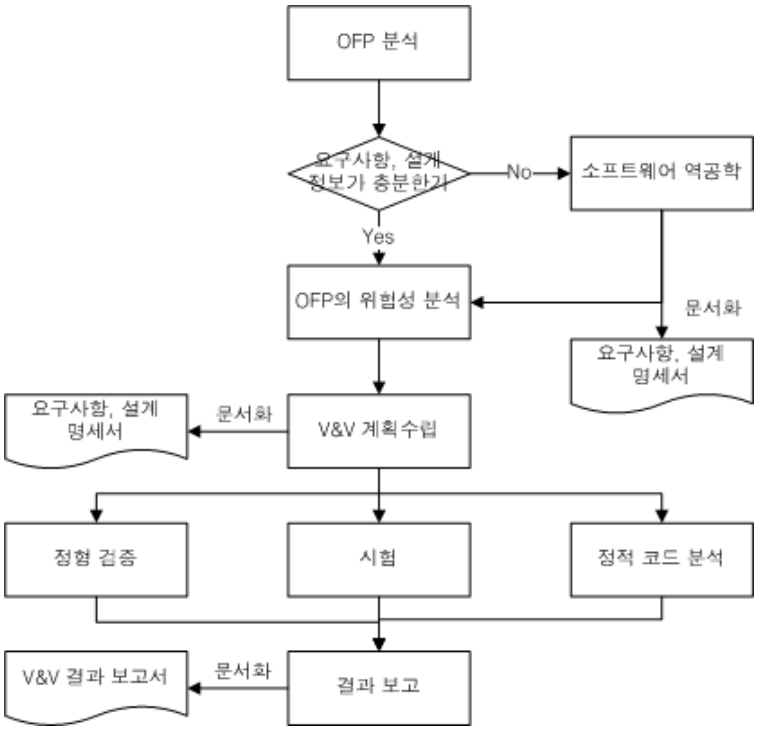
- HeliScope OFP(Operational Flight Program)
  - HeliScope Project의 무인 헬리콥터에서 동작
  - 실시간으로 동작하여 비행 제어를 담당



HeliScope 무인 헬리콥터

배경 연구

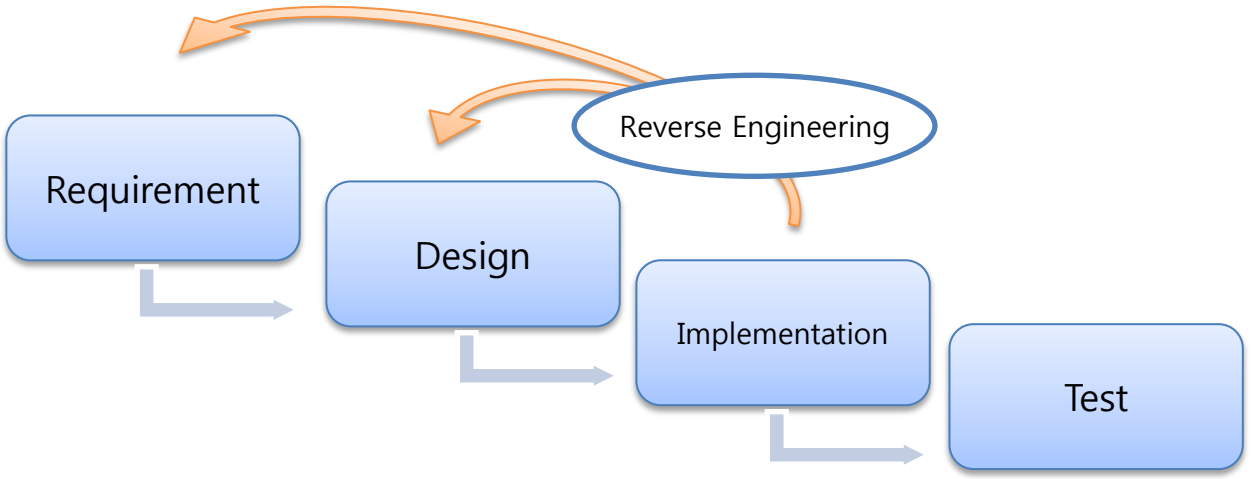
- HeliScope Project 비행 운용 프로그램(OFP)을 위한 검증 절차
  - OFP의 특징을 고려
    - Mission Critical 소프트웨어
    - 실시간, 내장형 소프트웨어
    - 개발이 완료된 소프트웨어
  - 검증 관련 표준을 참고
    - DO-178B 항공 소프트웨어 개발 지침
    - IEEE 1012-2004 소프트웨어 V&V 표준



배경 연구

- 소프트웨어 역공학
  - 소프트웨어의 구조와 사양 등을 분석하는 방법
  - 소프트웨어 유지보수, 재개발 등을 위한 목적으로 사용

- OFP는 개발이 완료된 소프트웨어
  - 요구사항, 설계 정보가 충분하지 않을 수 있음
  - 소프트웨어 역공학을 통하여 요구사항, 설계 정보 보완



배경 연구

- 소프트웨어 테스팅
  - 소프트웨어를 실행시켜 기능과 동작을 확인하는 기법

  - 구조적 테스트
    - 소프트웨어의 구조에 기반하여 테스트 케이스 작성
    - 구조에 기반하므로 소스 코드, 설계 정보를 이용
    - 오류 발견 확률이 높으나 기능을 만족하는지 확인이 어려움

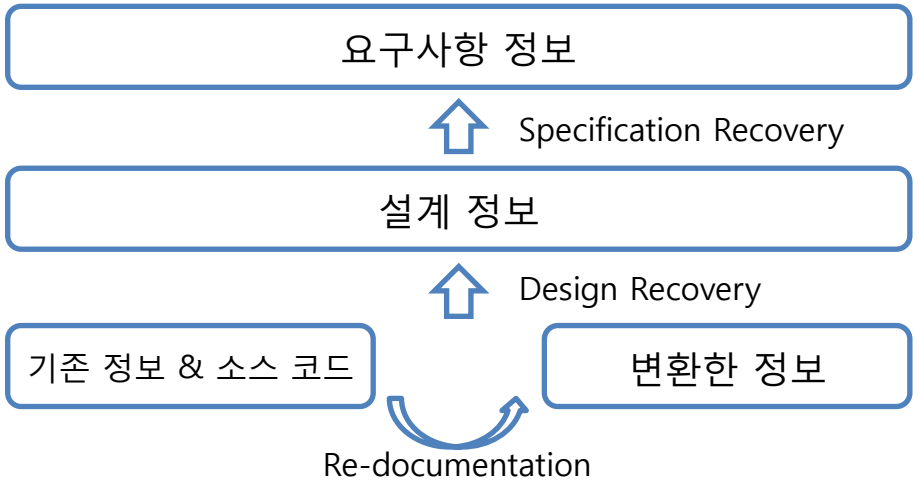
  - 기능적 테스트
    - 소프트웨어의 요구사항에 기반하여 테스트 케이스 작성
    - 요구사항에 기반하므로 요구사항 정보를 이용
    - 요구사항의 기능들을 확인할 수 있으나 일부분만을 테스트할 가능성

소프트웨어 역공학을 통한 HeliScope OFP 테스트 케이스 생성

적용 및 결과 – 소프트웨어 역공학

적용 및 결과 – 소프트웨어 역공학

- 소프트웨어 역공학 적용 절차
  - 기존 자료 수집 및 분석
    - 소프트웨어 역공학 적용 필요 여부를 판단
  - Re-documentation
    - 기존 정보들을 알아보기 쉬운 형태로 변환
  - Design Recovery
    - 설계 정보를 복원
  - Specification Recovery
    - 요구사항 정보를 복원



적용 및 결과 – 소프트웨어 역공학

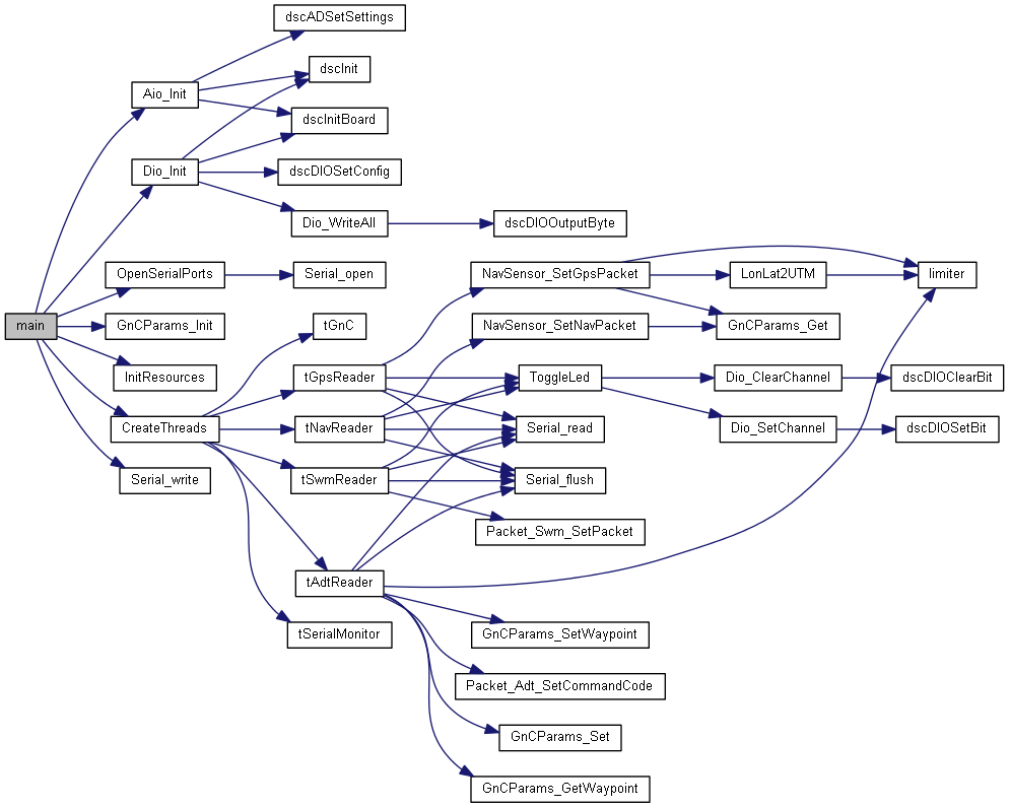
- Re-documentation
  - Data Description

변수 명	정의	Type
fdAdt	GCS 통신 시리얼 포트 접근 변수	int
fdDbg	Debug 모드 시리얼 포트 접근 변수	int
fdGps	GPS 센서 시리얼 포트 접근 변수	int
fdNav	Nav 센서 시리얼 포트 접근 변수	int
fdSwm	서보 상태 파악 및 컨트롤을 위한 시리얼 포트 접근 변수	int
semAdtReader	Adt 시리얼 포트로부터 데이터를 읽기 위한 세마포어	struct/sem_t
semGpsReader	GPS 시리얼 포트로부터 데이터를 읽기 위한 세마포어	struct/sem_t
semNavReader	Nav 시리얼 포트로부터 데이터를 읽기 위한 세마포어	struct/sem_t
semSwmReader	Swm 시리얼 포트로부터 데이터를 읽기 위한 세마포어	struct/sem_t

적용 및 결과 – 소프트웨어 역공학

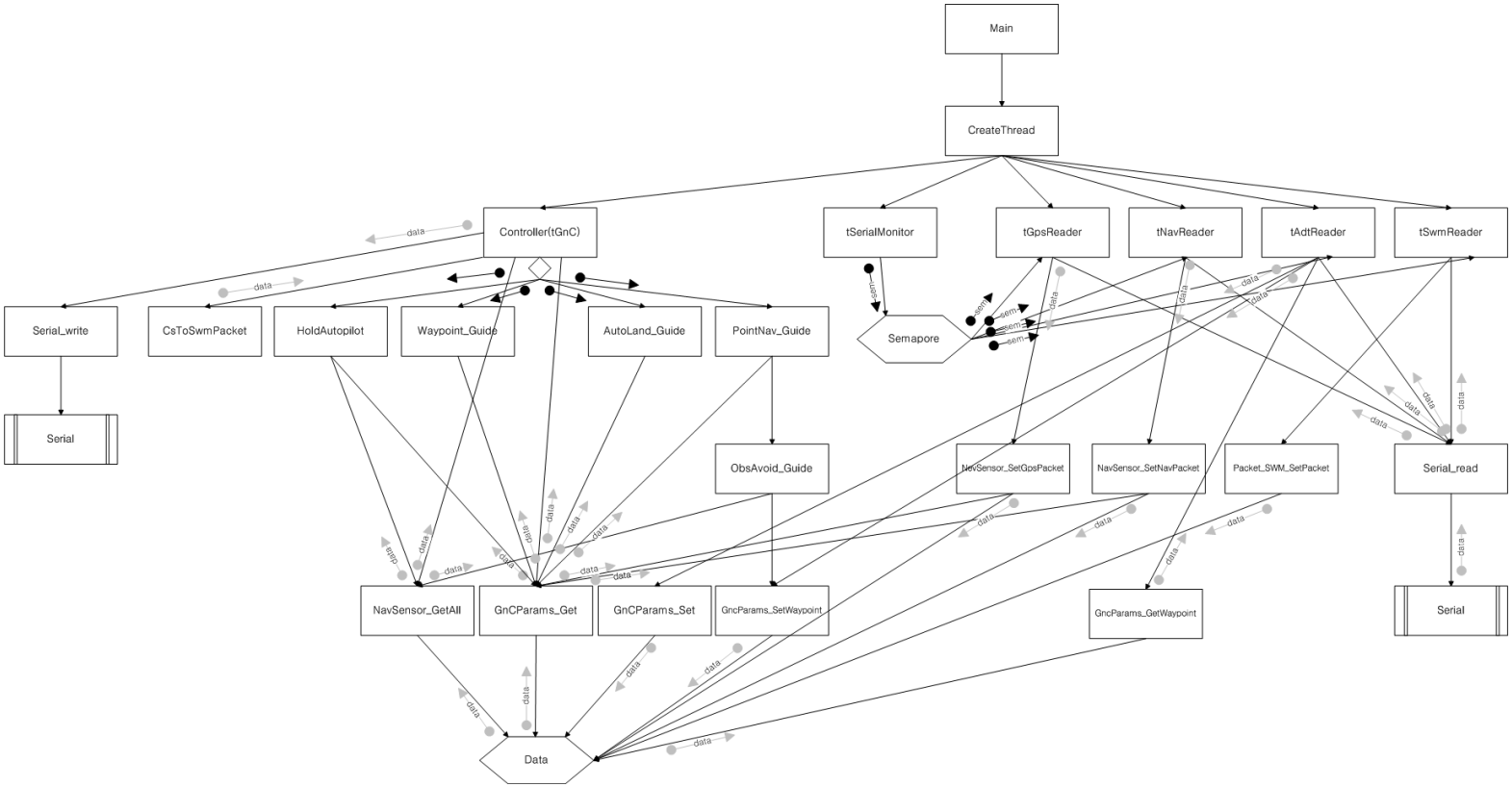
- Re-documentation(Cont'd)
  - Function List(Interface Description) & Call Graph

Aio_Close(fu)	void	()
Aio_Init(fu)	int	()
Aio_Read(fu)	int	(int,double *)
Aio_ReadAll(fu)	int	(double *)
Aio_ReadAllAvg(fu)	int	(double *)
Aio_ReadByteAll(fu)	int	(unsigned short *)
AutoLand_Guide(fu)	double	(double,double)
AutoLand_Init(fu)	void	(double,double,double)
CalcCRC(fu)	unsigned short	(unsigned short,unsigned char *,unsigned short)
CalculateBlockCRC32(fu)	unsigned long	(unsigned int,const unsigned char *)
CRC32Value(fu)	unsigned long	(const int)
CreateThreads(fu)	int	()
CsToSwmPacket(fu)	unsigned int	(double,double,double,double,unsigned char [])
DetectObstacles(fu)	int	(double,double,double)
Dio_ClearChannel(fu)	int	(BYTE,BYTE)
Dio_Close(fu)	void	()
Dio_Init(fu)	int	()
Dio_SetChannel(fu)	int	(BYTE,BYTE)
Dio_WriteAll(fu)	int	(BYTE,BYTE)
ddf(fu)	double	(double)
df(fu)	double	(double)
f(fu)	double	(double)
GetDistanceFromLeg(fu)	double	(int,double,double)
GetDistanceFromLeg(fu)	double	(int,double,double)
GetFuturePosition(fu)	void	(double,double *,double *)
GetFuturePosition(fu)	void	(double,double *,double *)
GetModeOfPN(fu)	int	(int,double,double)
GetModeOfPN(fu)	int	(int,double,double)
GetPathTarget(fu)	void	(int,double,double,double *,double *)
GetPathTarget(fu)	void	(int,double,double,double *,double *)
GetPushTarget(fu)	void	(int,double,double,double *,double *)
GetPushTarget(fu)	void	(int,double,double,double *,double *)
GnCParams_Get(fu)	double	(const unsigned char)
GnCParams_GetWaypoint(fu)	void	(const unsigned char,double *,double *)
GnCParams_Init(fu)	void	()
GnCParams_Set(fu)	void	(const unsigned char,double)
GnCParams_SetWaypoint(fu)	void	(const unsigned char,const double,const double)



적용 및 결과 – 소프트웨어 역공학

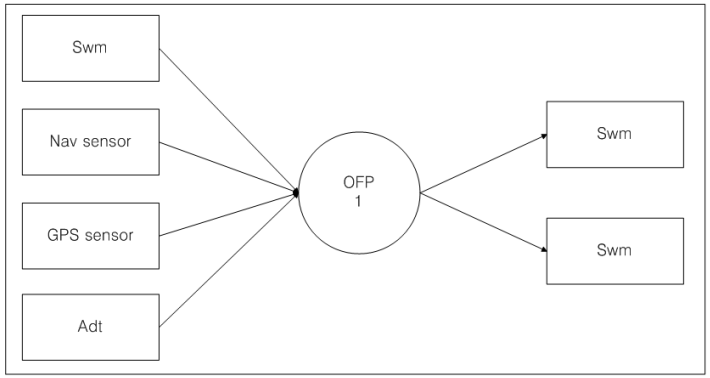
- Design Recovery
  - Structure-Chart



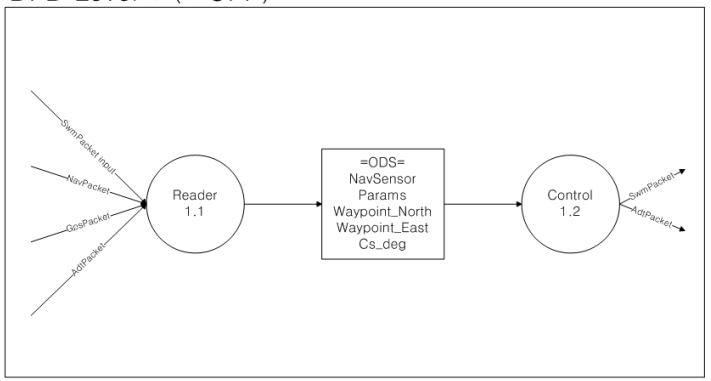
적용 및 결과 – 소프트웨어 역공학

- Design Recovery(Cont'd)
  - DFD(Data Flow Diagram)

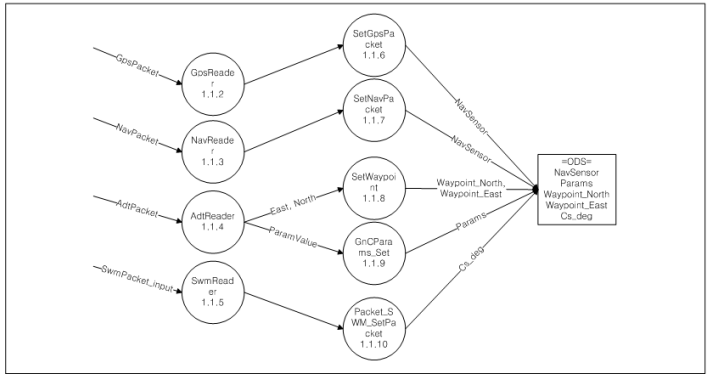
DFD Level 0



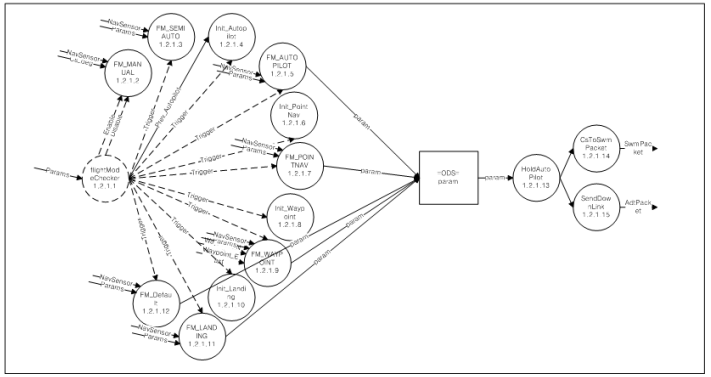
DFD Level 1 (1 OFP)



DFD Level 2 (1.1 Reader)



DFD Level 2 (1.2 Control)



적용 및 결과 – 소프트웨어 역공학

- Specification Recovery
  - Functionality of Processes

모듈 명	정의	입력/출력
SetGpsPacket	GPS 센서 값을 읽어 데이터 영역에 저장	GpsPacket/NavSensor
SetNavPacket	Nav 센서 값을 읽어 데이터 영역에 저장	NavPacket/NavSensor
SetWaypoint	GCS로부터 전송 받은 Waypoint 데이터를 데이터 영역에 저장	AdtPacket/ Waypoint_East, Waypoint_North
GnCParams_Set	paramValue값(입력 중 GPS 제외)을 데이터 영역에 저장	paramValue/ Params
Packet_SWM_SetPacket	Cs_deg값(3축 각도)을 데이터 영역에 저장	SwmPacket/Cs_deg
msgIdChecker	msgId값 확인(GCS의 커맨드 데이터의 타입 확인)	AdtPacket/msgId
flightModeChecker	다음 비행 상태를 위한 flight mode 확인	Params/

소프트웨어 역공학을 통한 HeliScope OFP 테스트 케이스 생성

적용 및 결과 – 소프트웨어 테스팅

적용 및 결과 – 소프트웨어 테스팅

구조적 테스트

- 구조에 기반(분기)
- 유닛 레벨의 테스트
- Statement 구조적 커버리지 측정

기능적 테스트

- 요구사항에 기반
- 시스템 레벨의 테스트
- 두 가지 방법으로 접근
  - 상태 전이 시스템 기반 테스트
  - 경계 값 기반 테스트

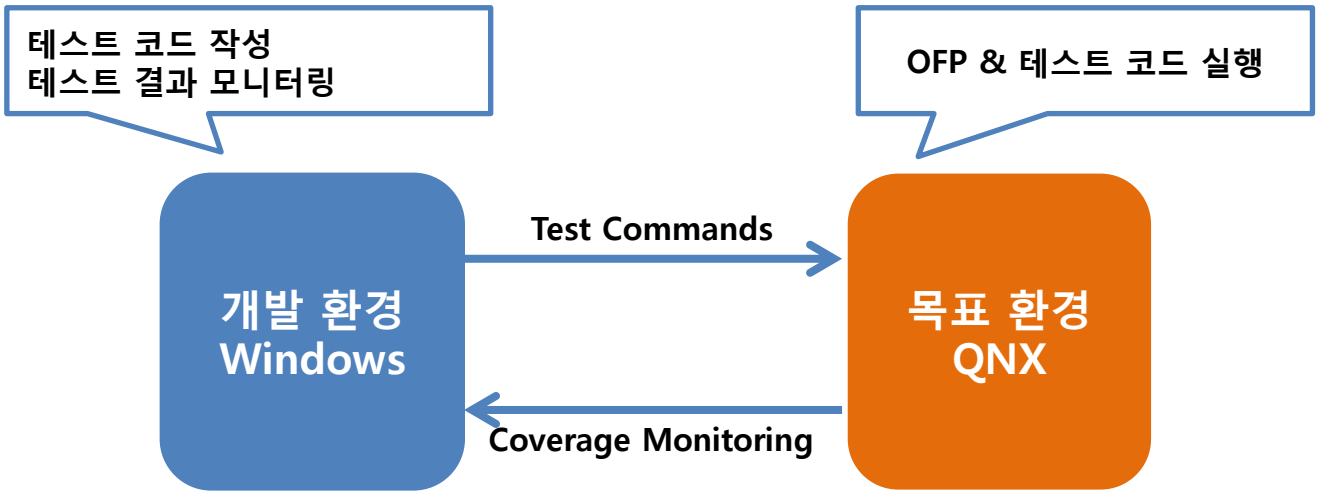
적용 및 결과 – 소프트웨어 테스팅

- 구조적 테스트
  - 테스트 케이스 작성(54개)

ID	모듈 명	정의
TC_S01	tSerialMonitor	semAdt == true
TC_S02	tSerialMonitor	semAdt == false
TC_S03	tSerialMonitor	semNav == true
TC_S04	tSerialMonitor	semNav == false
TC_S05	tSerialMonitor	semSwm == true
TC_S06	tSerialMonitor	semSwm == false
TC_S07	tSerialMonitor	semGps == true
TC_S08	tSerialMonitor	semGps == false
TC_S09	tAdtReader	msgId == ADT_AP_ALT
TC_S10	tAdtReader	msgId == ADT_AP_U

적용 및 결과 – 소프트웨어 테스팅

- 구조적 테스트
  - 테스트 환경 구성



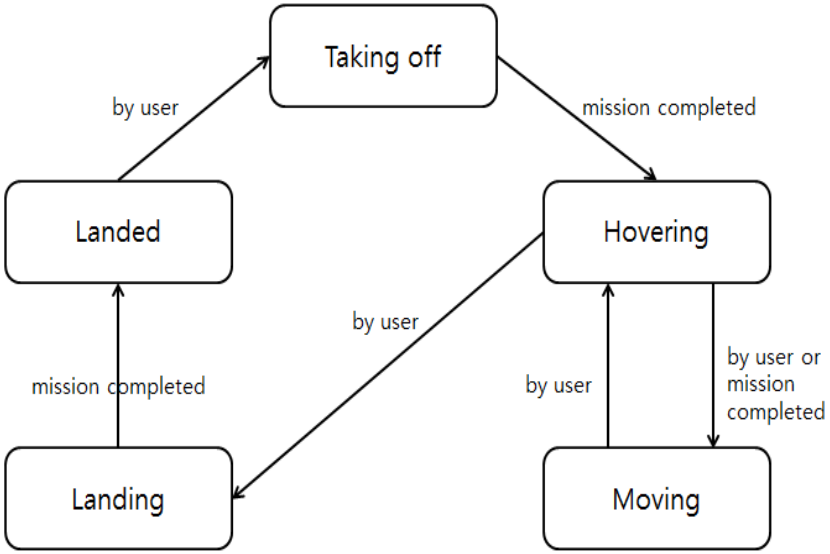
적용 및 결과 – 소프트웨어 테스팅

- 구조적 테스트
  - 테스트 수행(커버리지 측정 결과)

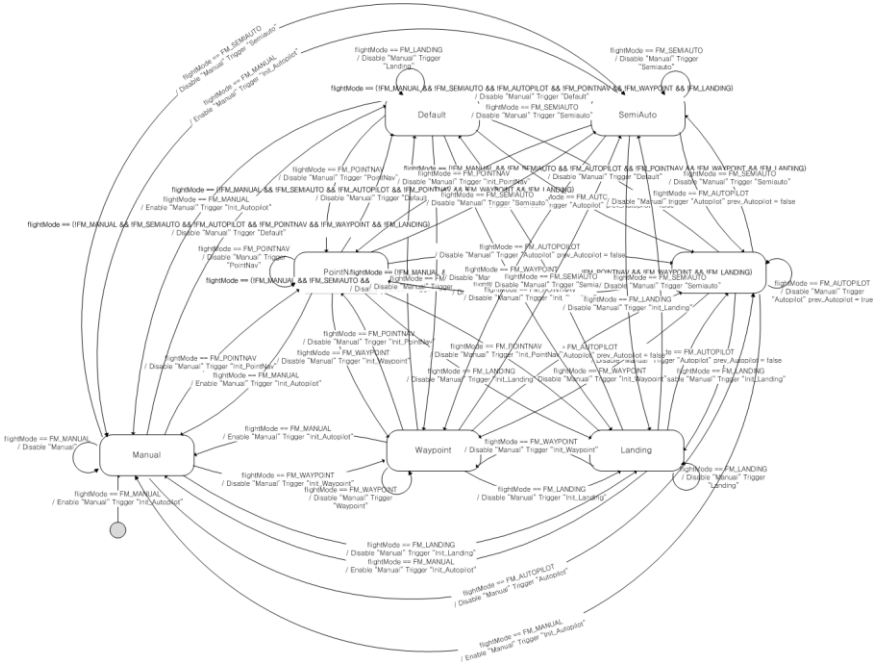
모듈 명	# of branch	# of Test Case	Statement Coverage
tGnC	18	11	99.47%
GnCAutopilot	0	1	100%
GnCAutoLand	16	17	100%
GnCPointNav	9	4	100%
GnCWaypoint	3	3	93.33%
GnCObsAvoid	20	4	86.26%
…	…	…	…
전체	-	54	94.79%

적용 및 결과 – 소프트웨어 테스팅

- 기능적 테스트 – 상태 전이 시스템 기반 테스트



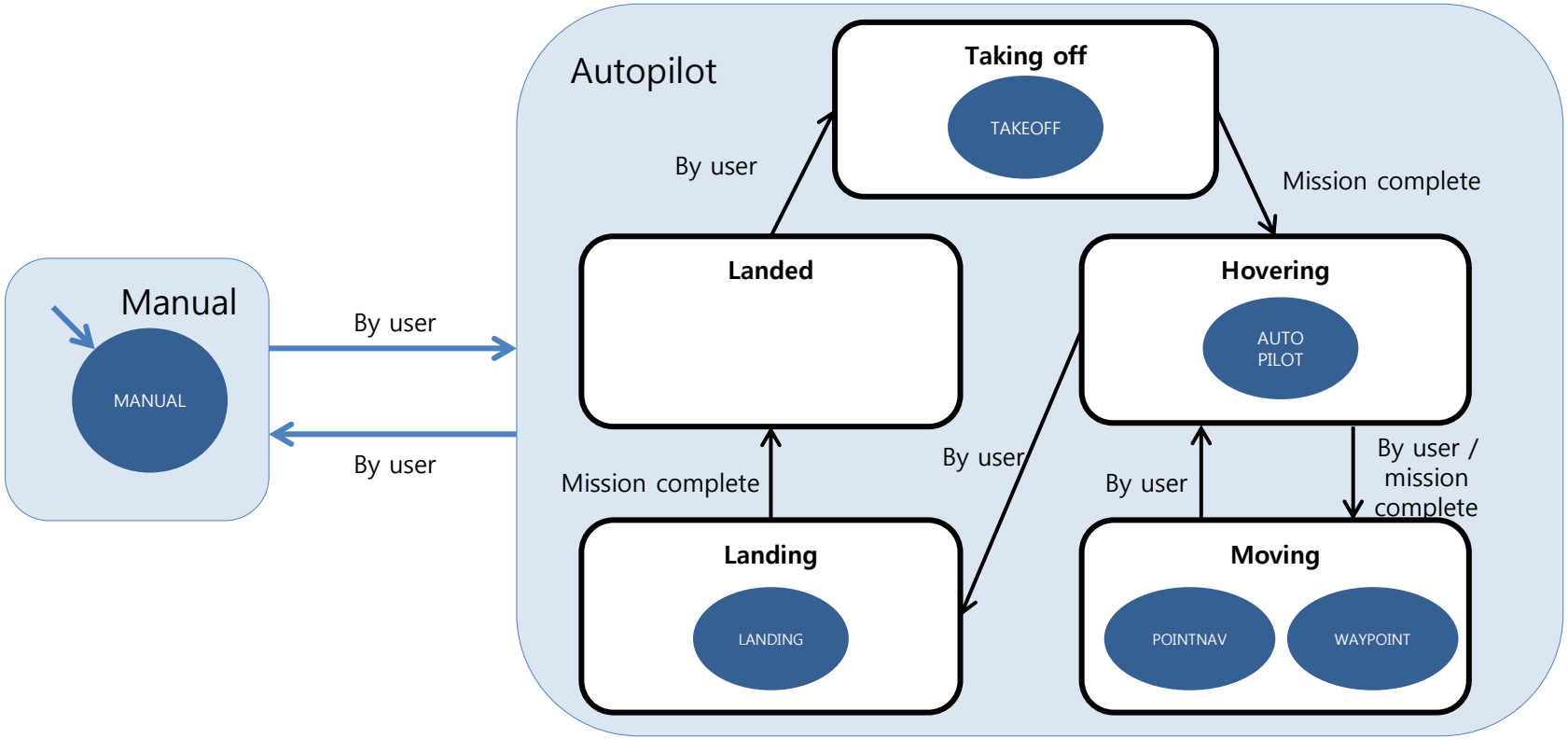
기존 문서에 존재하는 정보



역공학으로 복원된 정보

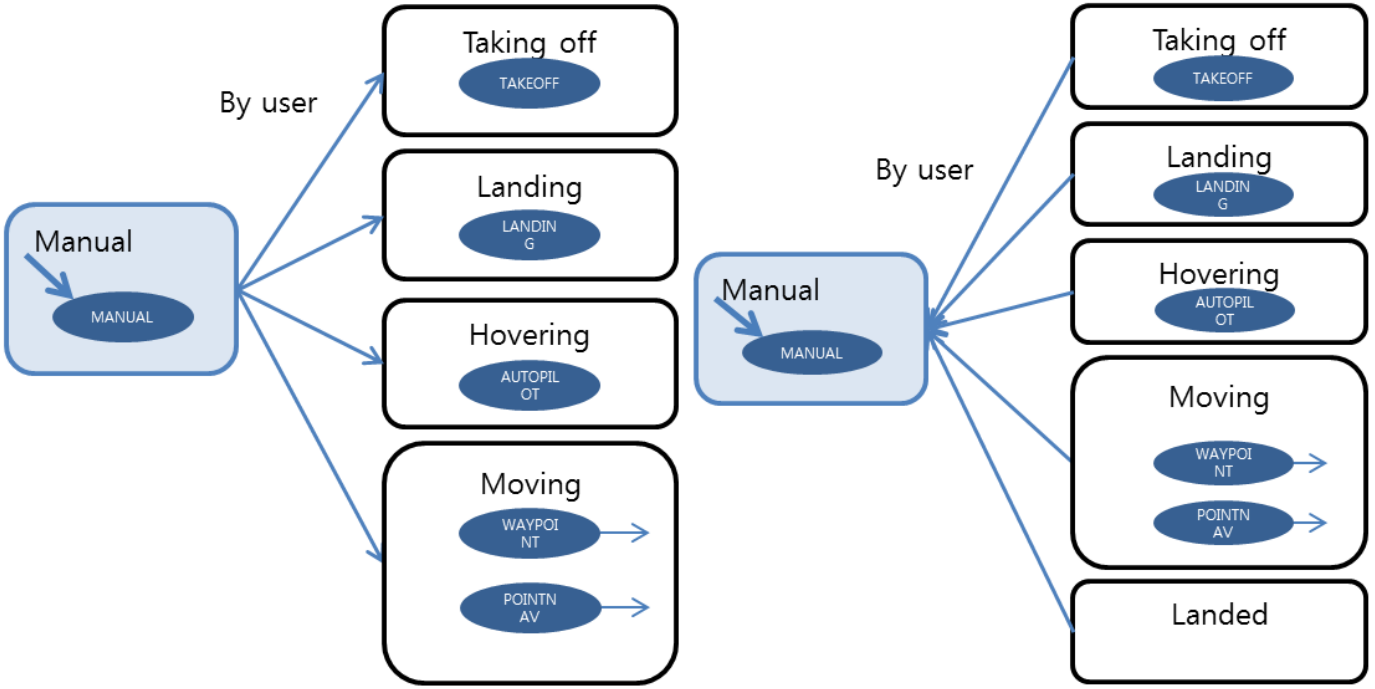
적용 및 결과 – 소프트웨어 테스팅

- 기능적 테스트 – 상태 전이 시스템 기반 테스트



적용 및 결과 – 소프트웨어 테스팅

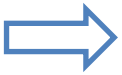
- 기능적 테스트 – 상태 전이 시스템 기반 테스트
  - 테스트 케이스 작성(18개)



적용 및 결과 – 소프트웨어 테스팅

- 기능적 테스트 – 경계 값 기반 테스트

- 유도제어법칙
  - Autopilot : GncAutopilot
    - 입력: U_mps, V_mps, H_mtr, Psi_rad
    - 출력: LonCyc_deg, LatCyc_deg, Col_deg, Rud_deg
  - PointNav : GncPointNav
    - 입력: East_mtr, North_mtr
    - 출력: U_mps, V_mps, Psi_rad
  - Waypoint : GncWaypoint
    - PointNav의 변형된 반복
  - 기타
    - AutoLand
    - Obstacle Avoidance



변수 명	타입	정의
msgId	unsigned short	ADT 패킷의 타입
flightMode	unsigned char	현재 비행모드
north	double	GPS 값
east	double	GPS 값
Altitude	double	고도 값
VN	double	속도 값
VE	double	속도 값
VD	double	속도 값
U	double	Forward 속도
V	double	Side 속도
…	…	…

적용 및 결과 – 소프트웨어 테스팅

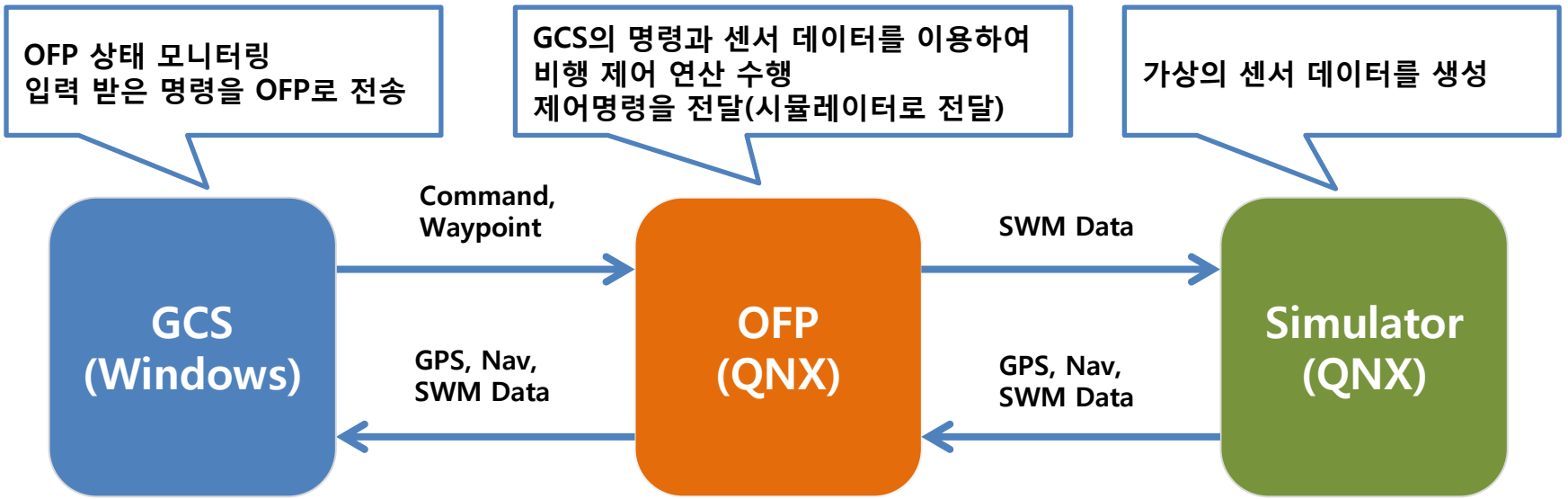
- Max boundary : 최대 경계 값
- Max+ : 최대 경계 값 초과의 값
- Min boundary : 최소 경계 값
- Min- : 최소 경계 값 미만의 값
- Nominal : 경계 값이 아닌 값

ID	데이터 명	값
TC_D00	msgId	0 (Min boundary)
TC_D01	msgId	-1 (Min-)
TC_D02	msgId	65535 (Max boundary)
TC_D03	msgId	65536 (Max+)
TC_D04	flightMode	0 (Min boundary)
TC_D05	flightMode	-1 (Min-)
TC_D06	flightMode	255 (Max boundary)
TC_D07	flightMode	256 (Max-)
TC_D08	Nav_Sensor[]	1.7e-308 (Min boundary)
TC_D09	Nav_Sensor[]	1.7e308 (Max boundary)

ID	데이터 명	값
TC_D10	msgId	0x0005 (Min-)
TC_D11	msgId	0x0006 (Min boundary)
TC_D12	msgId	0x0010 (Nominal)
TC_D13	msgId	0x0011 (Don't use)
TC_D14	msgId	0x0012 (Nominal)
TC_D15	msgId	0x0013 (Nominal)
TC_D16	msgId	0x0014 (Don't use)
TC_D17	msgId	0x0049 (Max-)
TC_D18	msgId	0x0050 (Max boundary)
TC_D19	msgId	0x0051 (Max+)

적용 및 결과 – 소프트웨어 테스팅

- 기능적 테스트
  - 테스트 환경 구성



소프트웨어 역공학을 통한 HeliScope OFP 테스트 케이스 생성

결론 및 향후 연구

결론 및 향후 연구

- 개발이 완료된 소프트웨어에 대한 검증
  - 소프트웨어 역공학을 적용하여 부족한 정보를 보완
  - 보완한 정보를 이용하여 소프트웨어 테스팅 적용

- 향후 연구
  - OFP의 요구사항 추가 보완
  - 시스템 레벨의 테스트 환경 구성 및 테스트 수행

Thank you