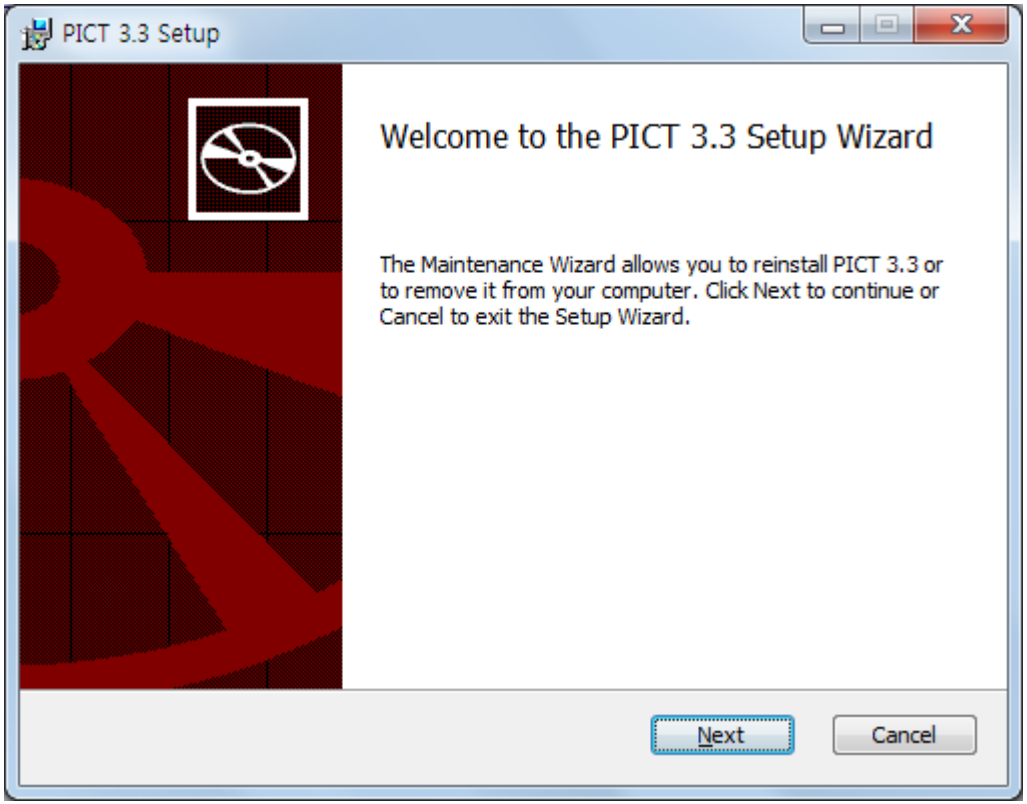
Pairwise Tool & Pairwise Test “NuSRS”

200511305 김성규
200511306 김성훈
200614164 김효석
200611124 유성배
200518036 곡진화

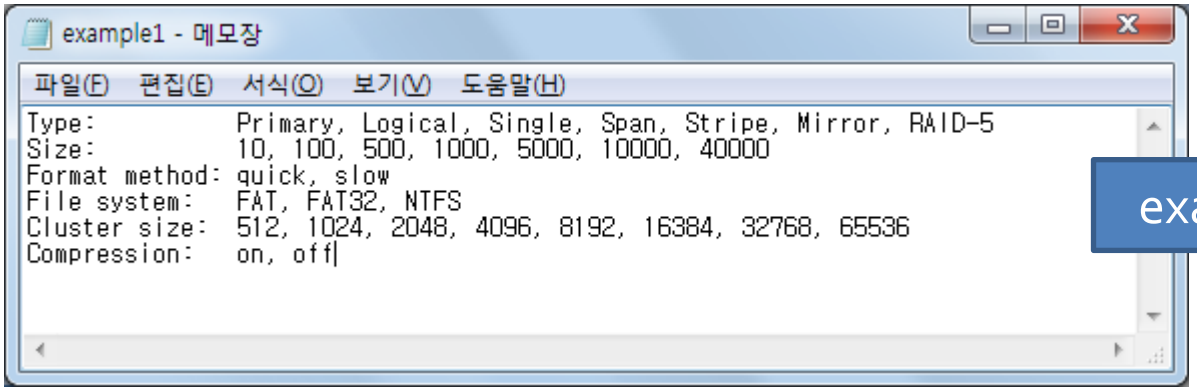
PICT

- Pairwise Tool - PICT
  - Microsoft의 Command-line 기반의 Free Software
  - www.pairwise.org에서 다운로드 후 설치



PICT

- Parameter, Constraint 정의



- Constraints 정의 가능
  - IF, Condition 조건
    - IF [File system] = "FAT" THEN [Size] <= 4096;
    - IF [Cluster size] in {512, 1024, 2048} THEN [Compression] = "Off";

PICT

- Pairwise 수행
  - 설치 폴더에서 수행 또는 환경 변수 추가
  - pict [정의 파일] > [결과 파일]
  - pict example1.txt > ex.xls

1	Type	Size	Format method	File systen	Cluster size	Compression
2	Mirror	10	quick	FAT	32768	off
3	RAID-5	10	slow	FAT32	512	on
4	Stripe	500	quick	NTFS	512	off
5	Span	1000	slow	NTFS	1024	on
6	Primary	100	quick	FAT32	16384	off
7	Single	1000	slow	FAT	8192	off
8	Primary	5000	slow	FAT	2048	on
9	RAID-5	40000	quick	NTFS	8192	on
10	Logical	10	slow	NTFS	65536	on
11	Span	100	quick	FAT	65536	off
12	Mirror	10000	slow	FAT32	65536	on
13	Logical	1000	quick	FAT32	512	off
14	Logical	40000	slow	FAT	4096	off
15	Single	1000	quick	NTFS	4096	on

ex.xls

PICT

❖ pict 명령어 옵션

```
C:₩Users₩Kestern₩Documents>pict example1.txt > ex.txt /o:6 /s

C:₩Users₩Kestern₩Documents>type ex.txt
Combinations:    4704
Generated tests: 4704
Generation time: 0:00:02

C:₩Users₩Kestern₩Documents>pict example1.txt > ex.txt /s

C:₩Users₩Kestern₩Documents>type ex.txt
Combinations:    331
Generated tests: 60
Generation time: 0:00:00
```

- /o:[숫자] ⇒ Combination 수 지정 (default : 2)
  파라미터 수 만큼 지정 시 모든 경우 출력
- /r:[숫자] ⇒ Pairwise 추출을 랜덤 (default : 0)
  옵션 없이 반복 수행 시 동일한 결과 획득
- /s ⇒ Combination, 생성된 test 수, 생성 시간

Functional specifications

- Expression is parsed to Logic structure
- Used symbol : |, &, +, -, =, !, <, >, <=, >=, :=, (,)
- Expression
  - Divide with operator and definition of variable (A=true)
  - Identify Boolean value of expression
  - If '(', ')' is occurred in expression, identify value
- Expression must be identified that value is false or true

Functional specifications

예) (f_SG1_LO_FLOW_Val_Out_t1 -f_SG1_LO_FLOW_Val_Out <=k_SG1_LO_FLOW_100ms_Rate) &
(f_SG1_LO_FLOW_Val_Out -k_SG1_LO_FLOW_Trip_Step >=k_SG1_LO_FLOW_SP_Lo_Lim) &
(f_SG1_LO_FLOW_Val_Out -k_SG1_LO_FLOW_Trip_Step <=k_SG1_LO_FLOW_SP_Hi_Lim)

문자열 변환 과정 (지난 발표 분석 참조)

Step 1. 복합문을 쪼갠다 : &, !, (,), | 로 나누기 - makeMember
Step 2. !가 미치는 구문을 찾아내어 상쇄시켜 중복 !연산을 최소화 - setTruth
Step 3. Step2에서 ! 영향을 변수로 저장했기 때문에 구문에서 ! 제거 - removeNot
Step 4. 괄호, 개행 문자 제거 - removeBracket, removeEnter
Step 5. 연산자에 대해 !을 수행 - convertTruth
Step 6. 나누어진 구문들을 비교 연산자를 기준으로 나눔 - setSubElement
Step 7. +, - 연산자를 기준으로 나눔 - divideSub

f_SG1_LO_FLOW_Val_Out_t1 / - / f_SG1_LO_FLOW_Val_Out / <=
k_SG1_LO_FLOW_100ms_Rate / & /f_SG1_LO_FLOW_Val_Out / -
k_SG1_LO_FLOW_Trip_Step / >= /k_SG1_LO_FLOW_SP_Lo_Lim) / &
f_SG1_LO_FLOW_Val_Out / - /k_SG1_LO_FLOW_Trip_Step / <=
k_SG1_LO_FLOW_SP_Hi_Lim)

Testable Units

Step 1. Identify Independently Testable Units

- 정해진 Symbol 사용 [Symbol]
- 올바른 변수 이름 사용 [Name]
- 올바른 Expression 형태 (A = true/false, A < B) [Exp]
  - 요구사항에서 추출

- ! 연산자 사용 [Not]
- 개행 문자 사용 [Enter]
  - removeNot, removeEnter 함수에서 추출

- 괄호 사용 [Level]
  - 요구사항, 함수에서 추출

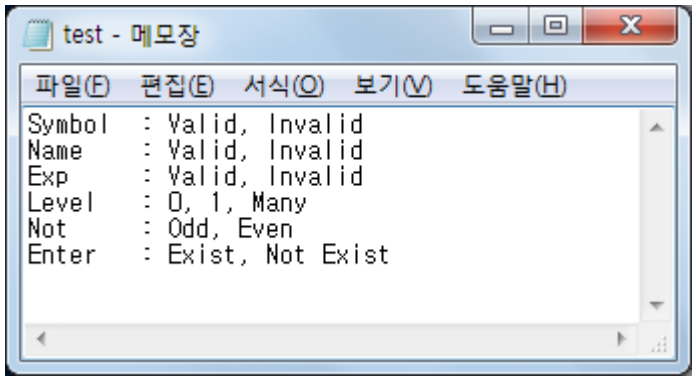
Representative Values

- Step 2. Identify Representative Values

	Value
Symbol	Valid, Invalid
Name	Valid, Invalid
Exp	Valid, Invalid
Not	Odd, Even
Enter	Exist, Not Exist
Level	0, 1, Many

Representative Values

- Step 2. Identify Representative Values
  - Pairwise 결과



pict test.txt > ex.xls /o:3

	A	B	C	D	E	F
1	Symbol	Name	Exp	Level	Not	Enter
2	Invalid	Valid	Valid	Many	Odd	Exist
3	Valid	Valid	Invalid	1	Even	Not Exist
4	Valid	Invalid	Valid	Many	Even	Not Exist
5	Invalid	Invalid	Invalid	Many	Odd	Not Exist
6	Invalid	Valid	Invalid	Many	Even	Not Exist
7	Valid	Invalid	Invalid	Many	Odd	Exist
8	Valid	Valid	Valid	0	Odd	Not Exist
9	Valid	Invalid	Valid	0	Even	Exist
10	Invalid	Invalid	Valid	1	Even	Not Exist
11	Invalid	Valid	Invalid	1	Odd	Exist
12	Invalid	Invalid	Invalid	0	Even	Not Exist
13	Valid	Invalid	Valid	1	Odd	Not Exist
14	Valid	Valid	Valid	1	Even	Exist
15	Invalid	Valid	Invalid	0	Even	Exist
16	Invalid	Invalid	Invalid	Many	Even	Exist
17	Valid	Invalid	Invalid	0	Odd	Exist
18	Valid	Invalid	Invalid	1	Odd	Exist
19	Invalid	Valid	Valid	0	Odd	Exist
20	Valid	Valid	Valid	Many	Odd	Not Exist

Testcase Specifications

- Step 3. Generate Testcase Specifications
  - Error Constraints
    - Symbol, Name, Exp 의 Invalid 시 Error

	Symbol	Name	Exp	Level	Not	Enter
1	Invalid	Valid	Valid	Many	Odd	Exist
2	Valid	Valid	Invalid	1	Even	Not Exist
3	Valid	Invalid	Valid	Many	Even	Not Exist
4	Invalid	Invalid	Invalid	Many	Odd	Not Exist
5	Invalid	Valid	Invalid	Many	Even	Not Exist
6	Valid	Invalid	Invalid	Many	Odd	Exist
7	Valid	Valid	Valid	0	Odd	Not Exist
8	Valid	Invalid	Valid	0	Even	Exist
9	Invalid	Invalid	Valid	1	Even	Not Exist
10	Invalid	Valid	Invalid	1	Odd	Exist
11	Invalid	Invalid	Invalid	0	Even	Not Exist
12	Valid	Invalid	Valid	1	Odd	Not Exist
13	Valid	Valid	Valid	1	Even	Exist
14	Invalid	Valid	Invalid	0	Even	Exist
15	Invalid	Invalid	Invalid	Many	Even	Exist
16	Valid	Invalid	Invalid	0	Odd	Exist
17	Valid	Invalid	Invalid	1	Odd	Exist
18	Invalid	Valid	Valid	0	Odd	Exist
19	Valid	Valid	Valid	Many	Odd	Not Exist

Testcase Specifications

❖ Step 3. Generate Testcase Specifications

	TestString
1	f_input1=false&!f_input2!=true
2	(f=_input1=true¦!f_input2=true)&f_input3=true
3	(f_input1=true&&!f_=input2==true)
4	!(!(!f_input1=true&!f_input2=false)\n&&!f_input3=false)
5	(f_input1=false&(f_input2=true)
6	(f_input1<>false&f_input2=true)¦input3=3
7	!(f_input1=true&f_input2=true)
8	!(f=_input1=false)¦f_input2=true
9	(f_input1==true&!f_input2=true)¦input3=false
10	(f_input=true¦(f_input2>=false)&input3=true)
11	((f_input1=false¦input2=true)&\n(f_input3=true&f_input4=true))\n
12	(f=_input1>=true¦!f_input2<=true)&f_input3!=true
13	!(!f_input1=true\n&f_input2=true)&!f_input3=true
14	f_input1=true\n&f_input2
15	(f_input!1=true\n&f_input2)?(f_input3=true&f_input4=true)
16	!f_input!1=true\n&f_input2
17	(!f_input1=true\n&f_input2)&!f_input>3=true
18	f_input1*true&\n!f_input2=true
19	((!f_input1=true&f_input2=true)&f_input3=true)

Testcase Code

❖ Step 4. Generate Testcase Code

	Symbol	Name	Exp	Level	Not	Enter
13	Valid	Valid	Valid	1	Even	Exist

```
public void testStringToLogic13() {
    String fullString =
    "!(!f_input1=true\n&f_input2=true)&!f_input3=true";

    stringToLogic s = new stringToLogic(fullString);
    Iterator<logicMember> it = s.lm.iterator();

    assertEquals(((logicMember)it.next()).element,"f_input1=true");
    assertEquals(((logicMember)it.next()).element,"|");
    assertEquals(((logicMember)it.next()).element,"f_input2 = false");
    assertEquals(((logicMember)it.next()).element,"&");
    assertEquals(((logicMember)it.next()).element,"f_input3 = false");

    it=s.lm.iterator();
```

Testcase Code

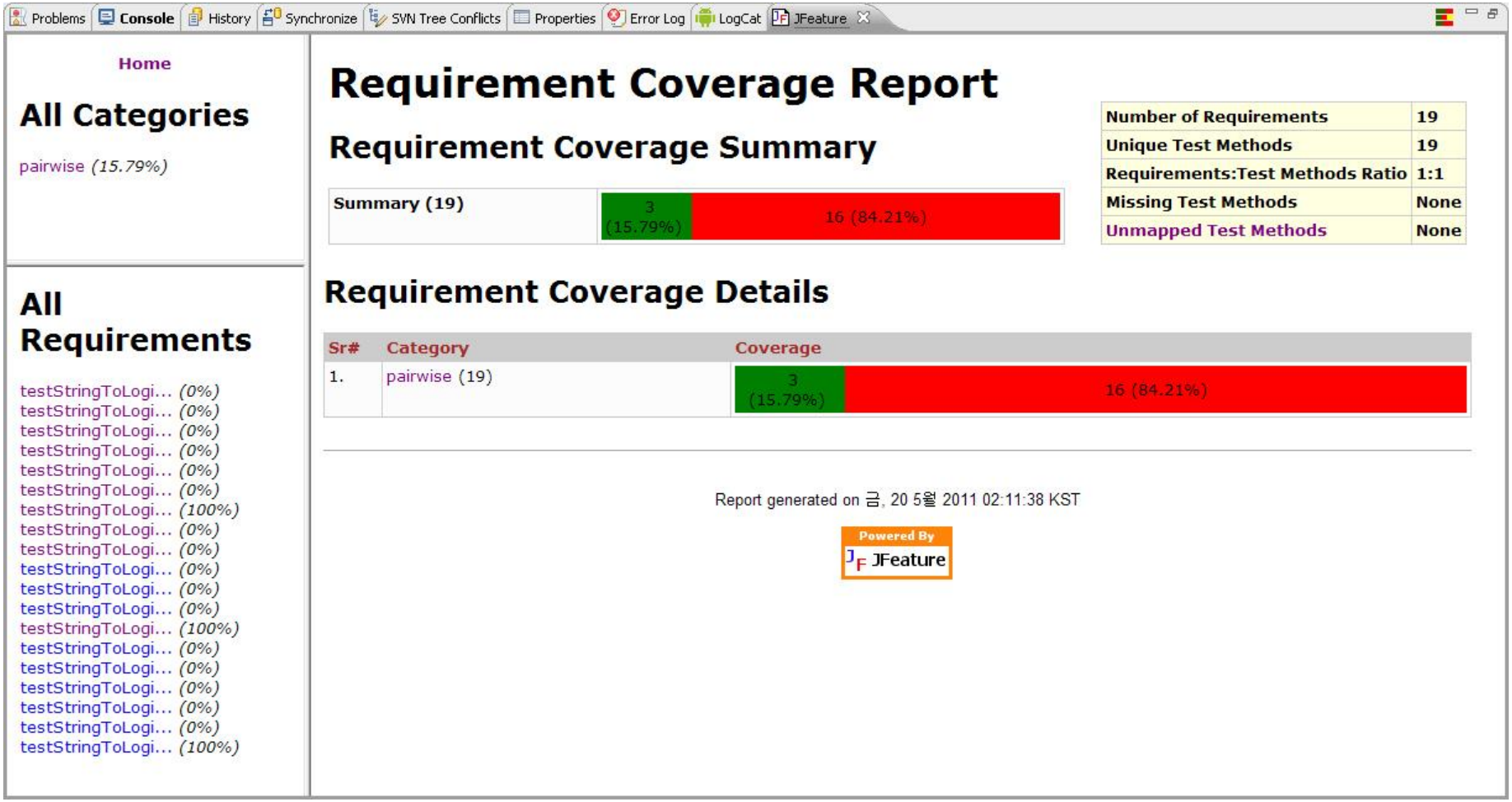
- Step 4. Generate Testcase Code

```
    assertEquals(((logicMember)it.next()).level,1);
    assertEquals(((logicMember)it.next()).level,1);
    assertEquals(((logicMember)it.next()).level,1);
    assertEquals(((logicMember)it.next()).level,0);
    assertEquals(((logicMember)it.next()).level,0);

    it=s.lm.iterator();

    assertEquals(((logicMember)it.next()).numNot,2);
    assertEquals(((logicMember)it.next()).numNot,1);
    assertEquals(((logicMember)it.next()).numNot,1);
    assertEquals(((logicMember)it.next()).numNot,0);
    assertEquals(((logicMember)it.next()).numNot,1);
}
```

Coverage Report



Coverage Report

Requirement Coverage Details

Sr#	Coverage Item	Coverage
1.	testStringToLogic_01 (1)	1 (100%)
2.	testStringToLogic_02 (1)	1 (100%)
3.	testStringToLogic_03 (1)	1 (100%)
4.	testStringToLogic_04 (1)	1 (100%)
5.	testStringToLogic_05 (1)	1 (100%)
6.	testStringToLogic_06 (1)	1 (100%)
7.	testStringToLogic_07 (1)	1 (100%)
8.	testStringToLogic_08 (1)	1 (100%)
9.	testStringToLogic_09 (1)	1 (100%)
10.	testStringToLogic_10 (1)	1 (100%)
11.	testStringToLogic_11 (1)	1 (100%)
12.	testStringToLogic_12 (1)	1 (100%)
13.	testStringToLogic_13 (1)	1 (100%)
14.	testStringToLogic_14 (1)	1 (100%)
15.	testStringToLogic_15 (1)	1 (100%)
16.	testStringToLogic_16 (1)	1 (100%)
17.	testStringToLogic_17 (1)	1 (100%)
18.	testStringToLogic_18 (1)	1 (100%)
19.	testStringToLogic_19 (1)	1 (100%)

END