

# 서비스 프로토타이핑?

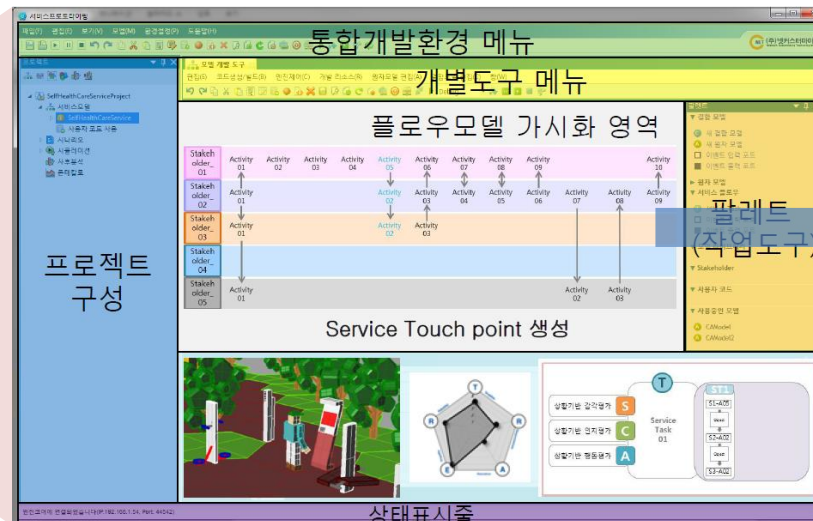
- 서비스 프로토타이핑 시스템

- 서비스를 실제 운용하기 전에 서비스의 UX 경험을 모델링하여 평가 및 개선하는데 도움을 주는 시뮬레이션 도구
- 시뮬레이션 지식이 없는 서비스 개발자가 서비스 프로토타이핑을 가능케 하도록 서비스 플로우 모델을 제공

- 서비스 플로우 모델

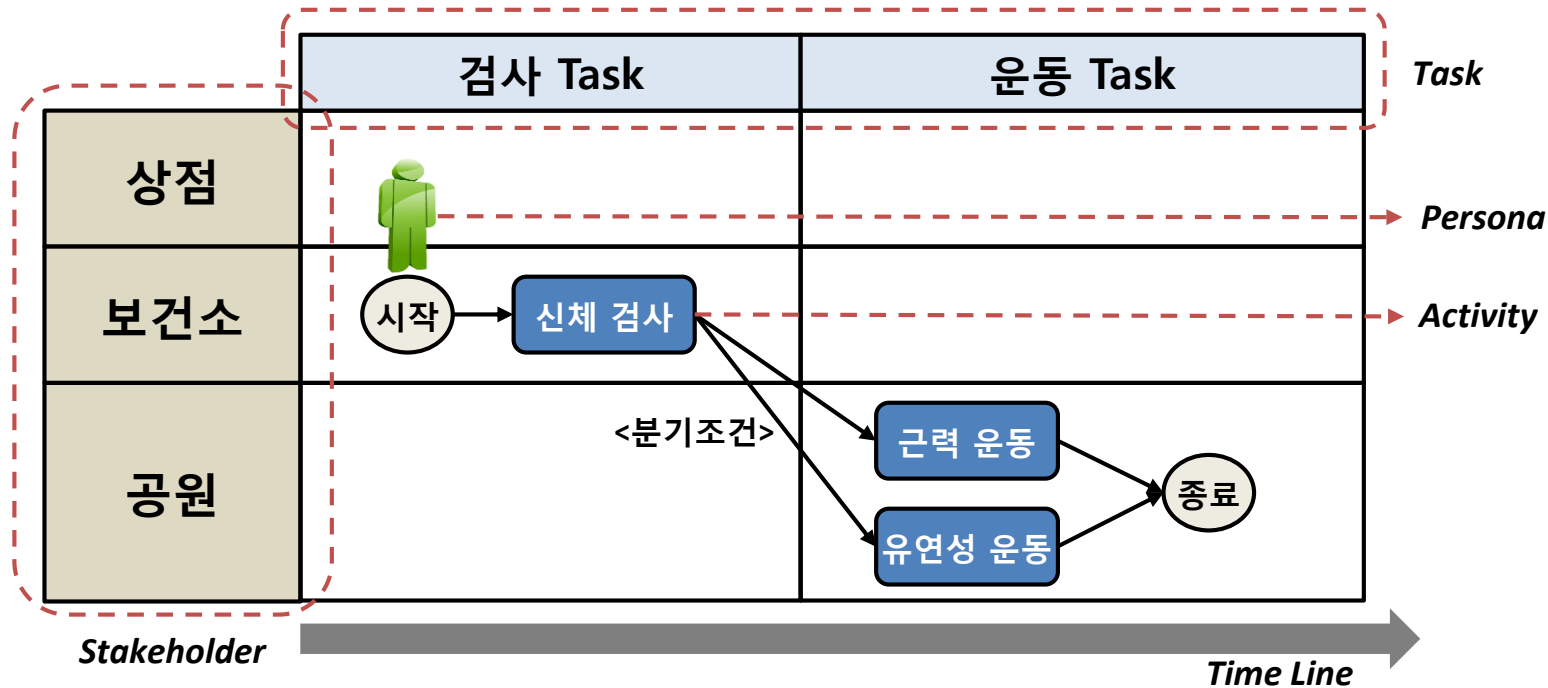
- 서비스 플로우 모델은 서비스를 사용하는 주체들의 서비스 흐름을 가시적으로 나타낼 수 있는 디지털 방식의 모델링 기술로 정의

서비스  
프로토타이핑  
시스템의 예



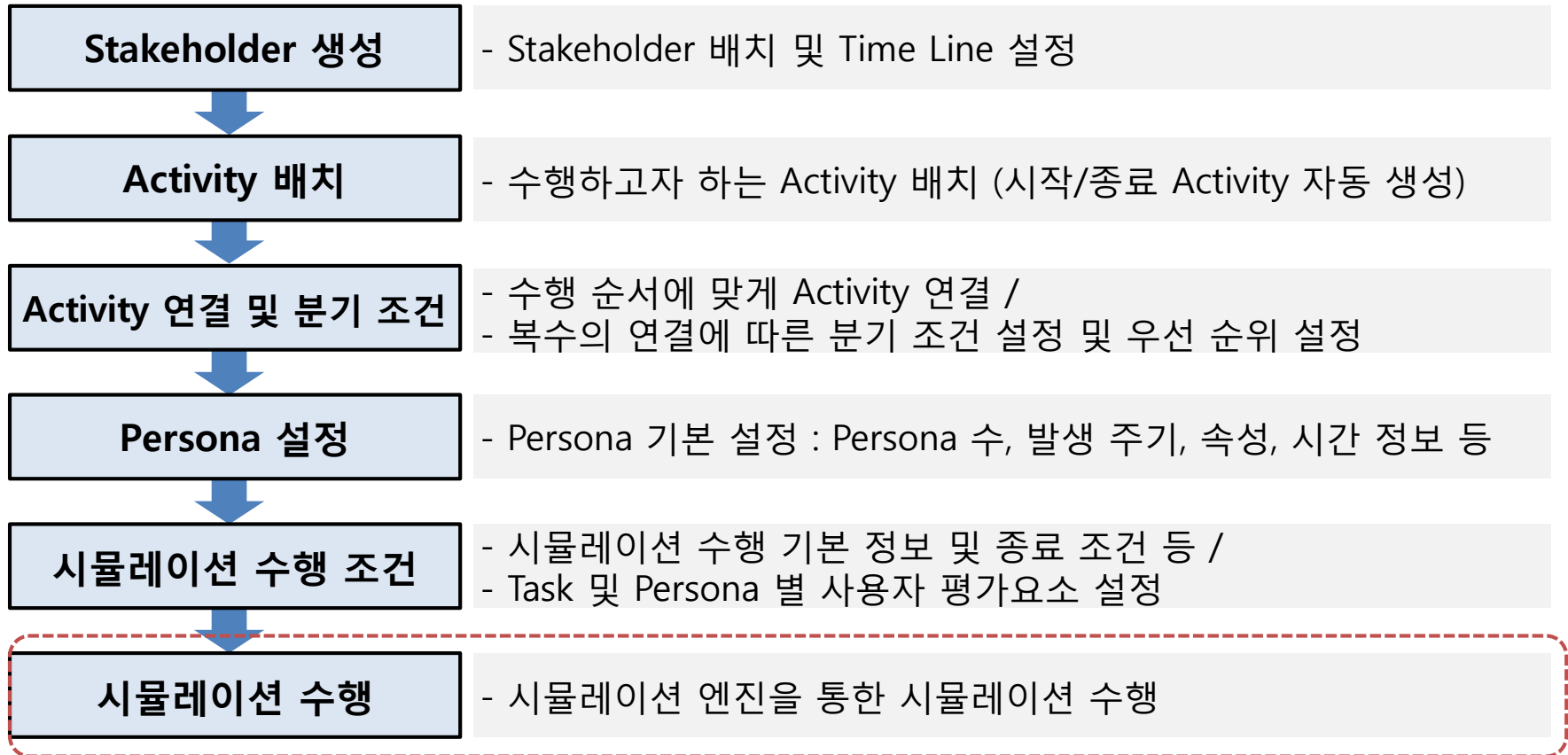
서비스 플로우 모델러

# 서비스 플로우 모델 구성요소



- **Stakeholder** : 서비스를 구성하고 있는 이해 관계자
- **Activity** : 서비스 사용 행위로, 가로 시간 축을 따라 생성
- **Activity Connection** : 각각의 Activity 사이를 연결하여 서비스의 흐름 가시화
- **Task** : 사용자의 서비스 평가 단위
- **Persona** : 서비스를 사용하는 주체

# 서비스 플로우 모델 수행 순서



생성된 서비스 플로우 모델을 시뮬레이션  
하기 위해 **DEVS 모델**로의 변환이 필요

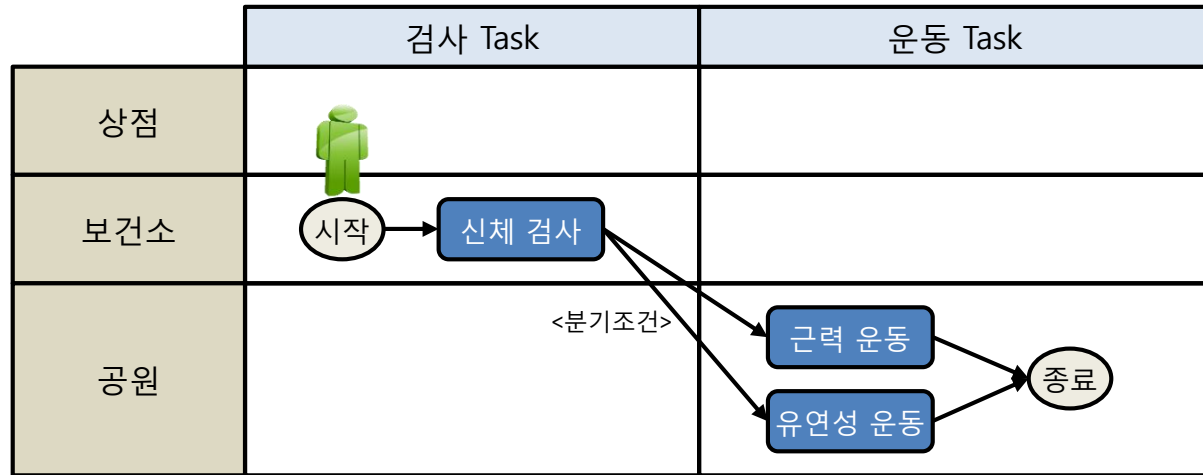


# DEVS 모델과의 관계 맵핑

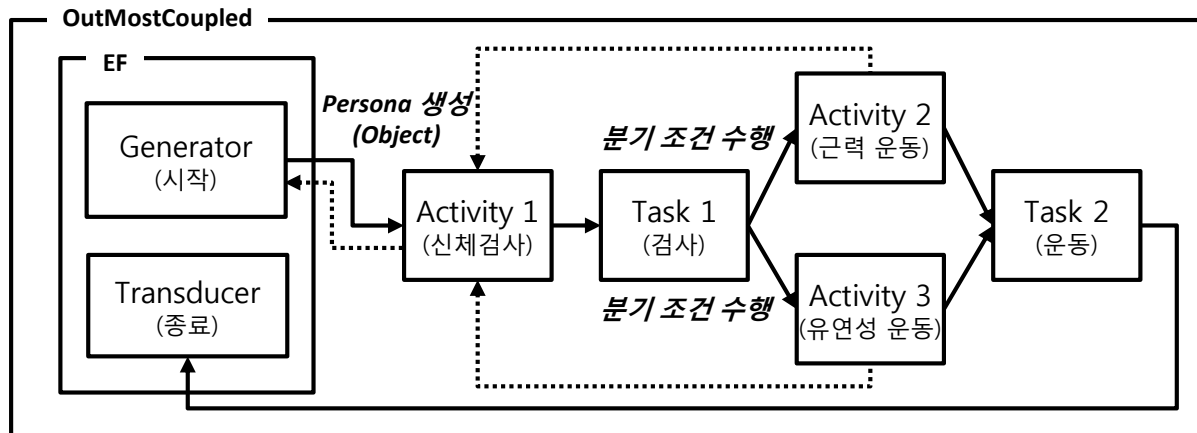
서비스 플로우	DEVS	설정 속성	비고
Activity (일반 모델)	Atomic Model	- 대기소의 수용량, 서비스 수 - Activity 원자 모델 구현	- 동일한 형태의 Atomic 모델
Activity (시작)	Atomic Model (Generator)	- Persona의 수, 발생 주기 설정	- 맨 앞의 Activity에 자동 추가
Activity (종료)	Atomic Model (Transducer)	- 종료 조건 설정	- 맨 뒤의 Activity에 자동 추가
Persona	Object (Message)	- Persona 내부 속성,	- 사용자 정의
Activity Connection	Coupling	- Activity 사이의 연결 관계	- 분기 조건 고려 필요
Stakeholder	-	-	- 서비스 종류 구분 단위
Task	Atomic Model	- 서비스 수행시간, 사용자 평가 지수	- 팔레트로부터 자동 생성

# DEVS 모델로의 변환

서비스 플로우 모델



DEVS 모델



# Activity 모델의 DEVS 변환

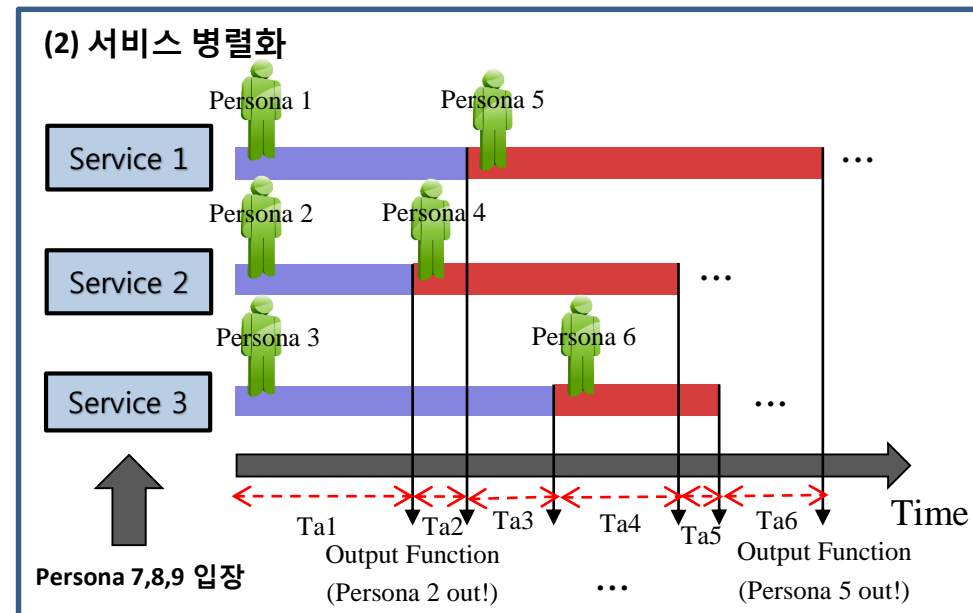
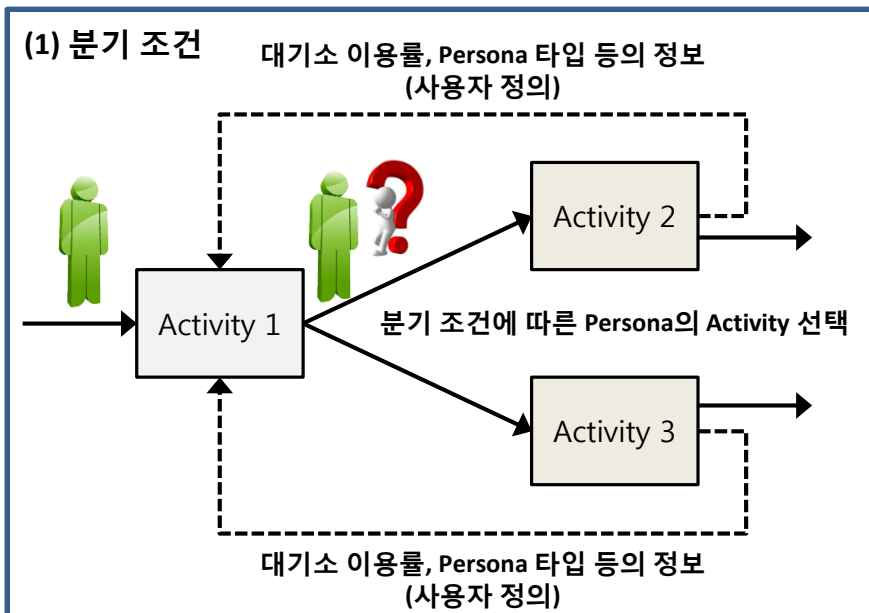
## • Activity 모델 변환 시 고려 사항

### (1) 분기 조건복수의 Activity 연결에 따른 분기 조건 설정

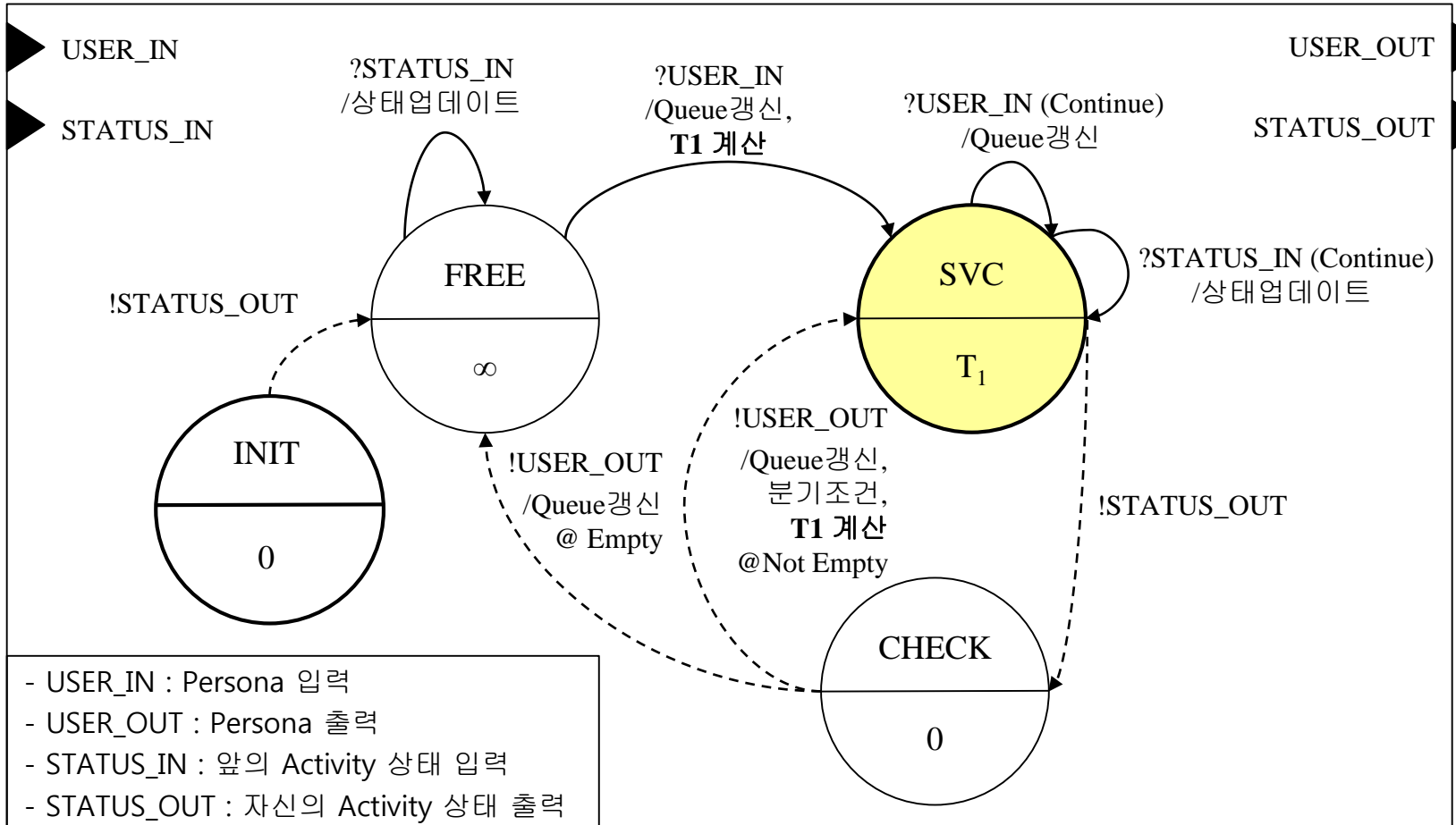
- 다음에 갈 수 있는 Activity가 여러 개일 때, Persona는 다음 Activity를 정해야 함
- 다음 Activity들의 현재 Status를 넘겨 받아 Persona가 판단
  - 대기소의 이용률, 사용 가능한 Persona 타입, Activity 환경 등의 Status

### (2) 서비스 병렬화

- 하나의 Activity에서 동일한 복수의 서비스를 수행하는 경우, Persona를 병렬적으로 처리할 필요가 있음
- 복수의 서비스 처리를 위해 적합한 시간 관리 방법 개발



# Activity DEVS 원자 모델 변환



# Task 모델 - 통계 수행

- 서비스에 대한 통계를 Task 모델에서 관리
  - 아래와 같은 통계를 (1) Activity 별, (2) Task 별로 나누어 수행

## ※ Persona 통계

- 최대, 최소, 평균 나이
- 남/여 비율
- Activity에 참여한 Persona 수

```

////////////////////////////////////
//Persona Statistic
std::vector<CPersona> vecPersona;

double m_maxAge;
double m_minAge;
double m_avgAge;
    
```

## ※ 만족도 통계

- 총, 최대, 최소, 평균 인지 만족도
- 총, 최대, 최소, 평균 감각 만족도
- 총, 최대, 최소, 평균 행동 만족도

```

////////////////////////////////////
// Satisfactory Evaluation
double m_totalRecScore;
double m_maxRecScore;
double m_minRecScore;
double m_avgRecScore;
    
```

```

double m_totalSensScore;
double m_maxSensScore;
double m_minSensScore;
double m_avgSensScore;
    
```

```

double m_totalBehScore;
double m_maxBehScore;
double m_minBehScore;
double m_avgBehScore;
    
```

## ※ 시간 통계

- 총, 최대, 최소, 평균 Wait Time (Start - Enter)
- 총, 최대, 최소, 평균 Service Time (Exit - Start)

```

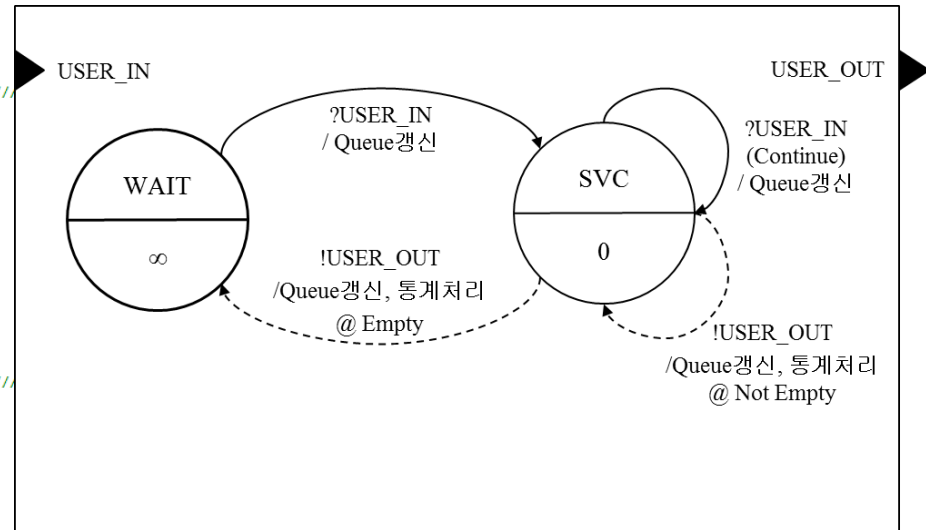
////////////////////////////////////
// Time Statistic
double m_totalWaitTime;
double m_maxWaitTime;
double m_minWaitTime;
double m_avgWaitTime;
    
```

```

double m_totalSvcTime;
double m_maxSvcTime;
double m_minSvcTime;
double m_avgSvcTime;
    
```

```

////////////////////////////////////
// Additional Information
std::map<std::string, ADD_INFO> mapAddInfo;
    
```



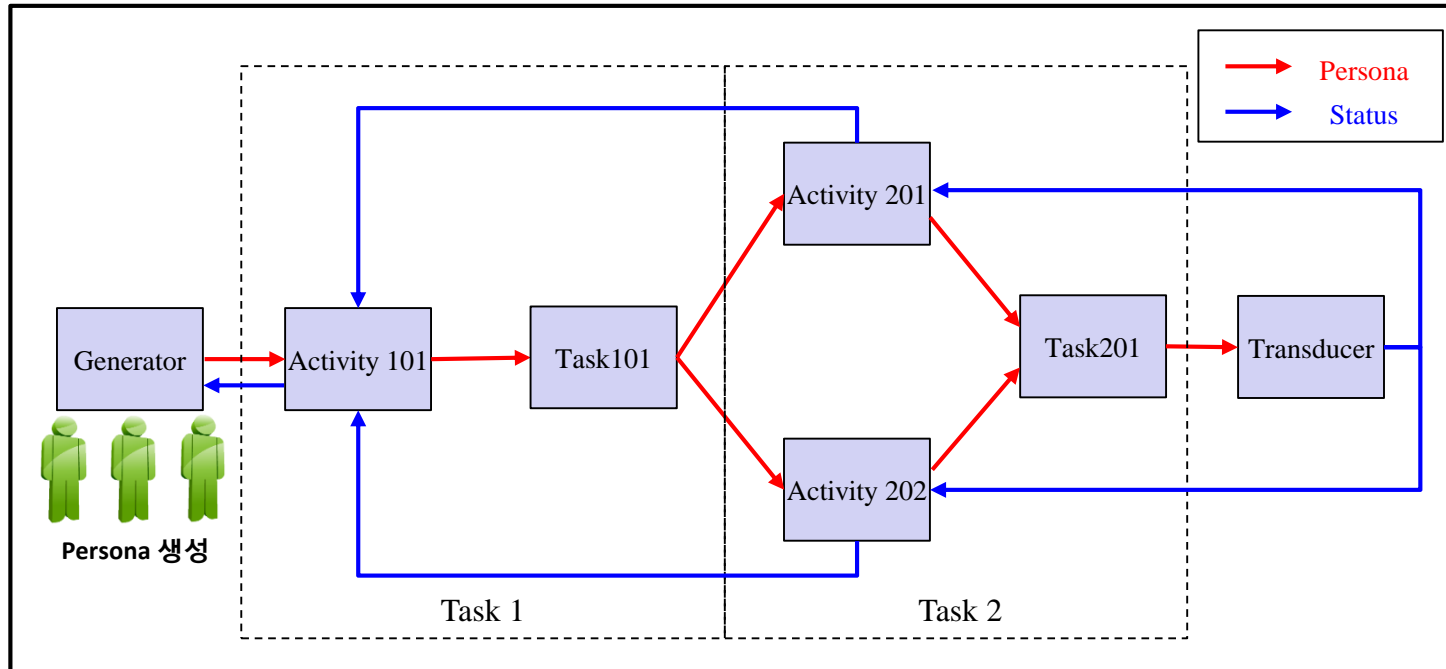
<Task 원자 모델 변환>



# 결론 및 추후 연구

- 시스템 프로토타이핑 시스템
  - 시뮬레이션 지식이 없는 서비스 사용자가 서비스를 모델링
  - 서비스 플로우 모델을 구성하고, 이를 시뮬레이션 하기 위해서는 DEVS 모델로의 변환이 필요
- 서비스 플로우 모델의 DEVS 변환
  - 각 Component에 대한 DEVS로의 맵핑
  - Activity 분기 조건 및 서비스 병렬화 해결
  - Task에서의 통계 처리에 대한 방법
- 추후 연구
  - 서비스 플로우 모델을 DEVS 모델로의 자동 변환하는 환경 구축

# 실험 디자인 및 결과



<Persona 속성>

	성별(Item[0])	나이(Item[1])
Persona 0	남	20~30
Persona 1	여	20~30
Persona 2	남	30~40
Persona 3	여	20~30
Persona 4	남	30~40
Persona 5	남	30~40

```
====Activity : 101 [Item
Total Persona Number: 4
Total Wait Time: 13
Max Wait Time: 6
Min Wait Time: 0
Avg Wait Time: 3.25
Total Svc Time: 15
Max Svc Time: 4
Min Svc Time: 3
Avg Svc Time: 3.75
```

```
====Task : 101 ====
Total Persona Number: 6
Total Wait Time: 16
Max Wait Time: 6
Min Wait Time: 0
Avg Wait Time: 2.66667
Total Svc Time: 25
Max Svc Time: 5
Min Svc Time: 3
Avg Svc Time: 4.16667
Total KSSQI Score: 40
Max KSSQI Score: 10
Min KSSQI Score: 5
Avg KSSQI Score: 6.66667
Total Emo Score: 24
Max Emo Score: 6
Min Emo Score: 3
Avg Emo Score: 4
```