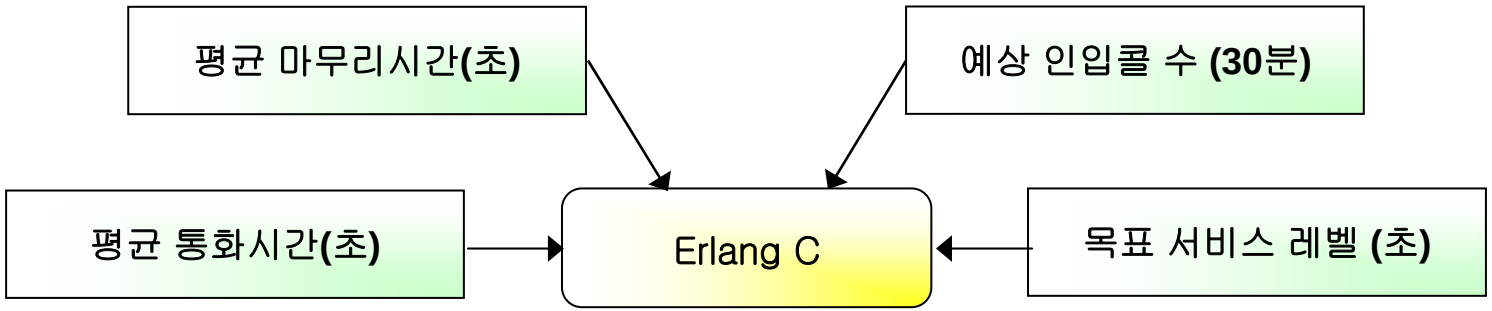
Understanding
Staff Calculation & Queuing Dynamics
상담원 계산 / 대기 행렬 이해

Erlang C

- Erlang C (created by A. K. Erlang)
  - 목표 서비스 레벨의 달성을 위해 필요한 상담원수를 산출하기 위한 공식

※ Erlang
- One hour of telephone traffic in an hour of time.
- 회선이 한시간 동안 120분 traffic 용량을 처리할 수 있으면 2 Erlangs

- Erlang C의 4가지 변수



Erlang C *cont'd*

- Erlang C 공식

$$P(>0)=\frac{\frac{A^N}{N!}\frac{N}{N-A}}{\sum_{x=0}^{N-1}\frac{A^x}{x!}+\frac{A^N}{N!}\frac{N}{N-A}}$$

- A = total traffic offered in erlangs
- N = Number of servers in a full availability group
- P(>0) =probability of delay greater than 0
- P = probability of loss

- P(>0), 즉 caller가 상담원과 즉시 연결되지 못하고 0보다 더 기다리게 될 확률

※ A. K. Erlang.

A Danish engineer who worked for the Copenhagen Telephone Company in the early 1900s and developed Erlang B, Erlang C and other telephone traffic engineering formulas.

Erlang C *cont'd*

- Erlang C Program의 Input / Output

Average Talk Time (Sec.)	150	Calls per Half-Hour	1000
After-Call Work Time (Sec.)	30	Service Level Objective (Sec.)	20

```
Average Talk Time (Sec.)       : 150
After-Call Work Time (Sec.)    : 30
Calls per Half-Hour            : 1000
Service Level Objective (Sec.): 20
```

TSRs	P(0)%	ASA	DLYDLY	Q1	Q2	SL%	OCC%	TKLD
101	88.3	159.0	180.0	88.3	100.0	21	99	171.7
102	77.7	69.9	90.0	38.9	50.0	38	98	122.2
103	68.1	40.8	60.0	22.7	33.3	51	97	106.0
104	59.4	26.7	45.0	14.8	25.0	62	96	98.2
105	51.6	18.6	36.0	10.3	20.0	70	95	93.6
106	44.6	13.4	30.0	7.4	16.7	77	94	90.8
107	38.4	9.9	25.7	5.5	14.3	82	93	88.8
108	32.8	7.4	22.5	4.1	12.5	87	93	87.4
109	28.0	5.6	20.0	3.1	11.1	90	92	86.4
110	23.7	4.3	18.0	2.4	10.0	92	91	85.7
111	20.0	3.3	16.4	1.8	9.1	94	90	85.1

- 서비스 레벨이 82% 인 상담원 수 107명 산출

Erlang C *cont'd*

Erlang C의 기본가정

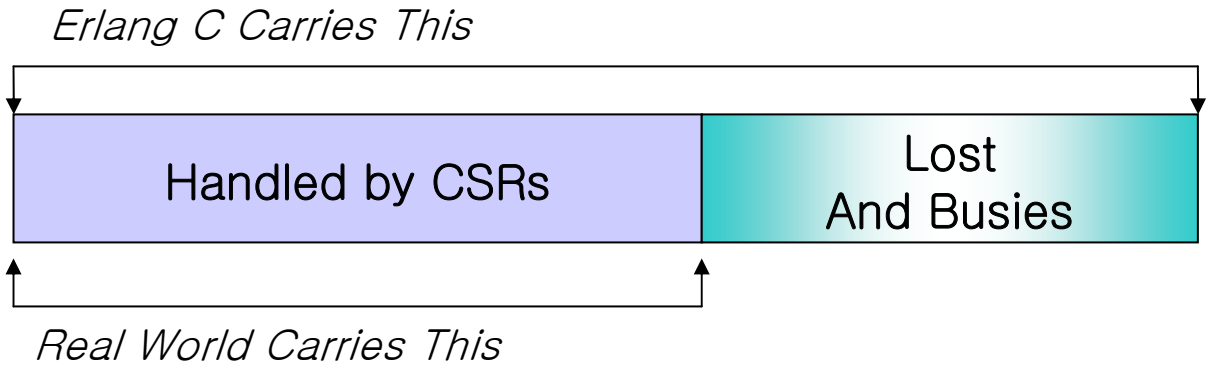
- 모든 인입콜은 동일한 종류 (single call type)
- 일단 Queue에 들어온 콜은 Abandon되지 않음
  - ➔ 모든 call은 접수되고(회선이 무한),
    모든 caller들은 서비스를 받을 때까지 기꺼이 기다림
- busy signal 없음
- 상담원들은 First-income, First-served basis에 입각하여 콜 처리
- 상담원 능력이 일정함
- 결근, 점심, 휴식 등의 상담원 부재 요인 배제

⇒ *결과적으로 Erlang C는 overstaff 산출*

- Erlang C의 변이
  - Pipkins의 Merlang
  - Hill's B formula

Erlang C *cont'd*

- Erlang C가 Overstaffing을 초래하는 이유



- *"그럼에도 불구하고.. 왜 Erlang C를 이용하는가?"*

1. Service Level을 충족시키기 위해 Erlang C를 사용할 때, lost calls와 busy signal은 별로 문제가 되지 않는다.
2. Lost calls와 busy signal을 고려하면, 오히려 understaffing을 초래하게 된다. (사실 caller들은 높은 재시도율을 보이고 있으며 결국에 통화를 하게 된다.)
3. Erlang C보다는 평균 통화시간, 평균 wrap-up time, 콜량 같은 다른 투입요소의 부정확성이 staffing의 정확도를 떨어뜨린다.
4. 약간의 overstaffing은 안전망으로서의 역할을 한다. (대부분의 콜센터는 상담원 이직과 대체인력 채용과 훈련으로 인해 인력을 효율적으로 사용하고 있지 못하다.)

Anton Consulting

Erlang C *cont'd*

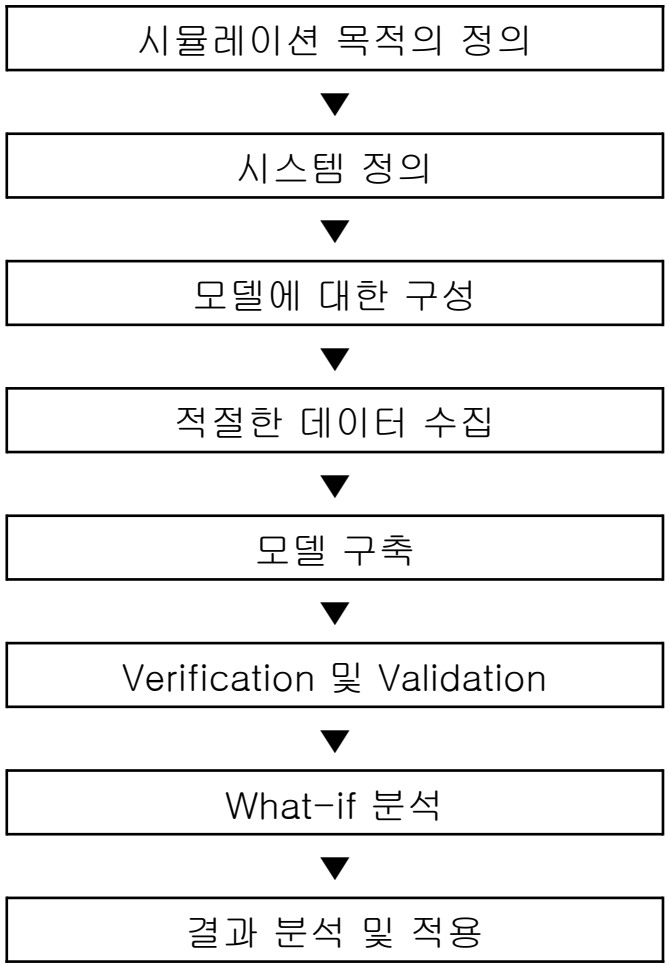
장 점 – simplicity	단 점 – accuracy
포기콜이나 busy signal이 최소화된 good service 수준에서는 정확 쉽고 빠르게 사용할 수 있으며, 폭 넓고 다양한 자료들로부터 소프트웨어 형태로 이용이 가능 콜센터의 tradeoffs를 잘 표현 (예, 서비스레벨이 올라갈 경우, occupancy 는 내려간다.) 거의 모든 Workforce management 소프트웨어 프로그램의 staffing을 계산하는 기본 공식	▪ 포기콜 및 busy signal이 없음 콜 인입 가정의 단순 – 일정 기간 동안 콜은 일정하게 들어오고, 콜량은 일정 수준 이상으로 늘거나 줄지 않는다고 가정 (폭주 없음) 예측 기간동안 콜을 응대하는 상담원의 수는 고정되어 있다고 가정 그룹 내의 모든 상담원은 해당 그룹에 주어진 모든 콜을 응대할 수 있다고 가정

Simulation

- Simulation 이란
  - 현실문제를 반영하는 모형을 만들어 실험을 함으로써 현실문제를 이해하고 여러가지 대안의 결과를 예측하는 방법

- 상담원 산출 simulation
  - 상담원 투입의 변화에 따른 콜센터 운영의 결과를 살펴 봄
  - 목표하는 수치에 맞는 상담원 투입 인원 조정해 봄으로써 적정 인원을 예상

Simulation 프로세스



- Verification(검사)
  - 시뮬레이션의 경험적 모형과 이것을 수행하기 위해 짜여진 컴퓨터 프로그램을 비교하는 것
  - 즉, 프로그램에 잘못은 없는지, 모형의 논리적 구조나 모수들이 정확하게 컴퓨터 프로그램에 반영되었는지 검사
- Validation(확인)
  - 시뮬레이션 모형이 현실문제와 일치하는지를 조사하는 것
  - 이 과정은 보통 반복적으로 수행
  - 즉, 먼저 모형을 만들고 이 모형과 현실문제를 비교하여 차이점이 있으면 모형을 수정하여 현실문제에 접근하도록 작성

Simulation의 장단점

장 점	단 점
Overflow, 그룹간의 공통적 업무, skill-based routing등과 같은 다양한 변수들에 대한 가정을 프로그램화 할 수 있음 분실된 콜이나 통화중 신호 등이 반영 각 콜센터의 환경에 맞추어 프로그래밍이 가능	구현과 활용에 시간이 소요되고, 상대적으로 전문적인 사용자가 요구됨 단독으로 구성된 tool이기 때문에, 예측이나 staffing 모듈과 통합되지 않음 단일 Erlang C 소프트웨어보다 고가

Erlang C vs. Simulation

- 무엇을 사용해야 하는가?

Erlang C	Simulation
좋은 서비스 레벨을 갖춘 단순한 업무 환경일 때	복잡한 routing 성능을 요할 때

현실을 완벽하게 예측하는 공식이나 프로그램은 존재하지 않는다.

각 콜센터의 상황에 따라 그에 맞는 tool을 사용해야 한다.

The Laws of Call Center Nature

- Occupancy, Service Level & 상담원 그룹 크기

"Occupancy(근무강도)와 service level은 반비례관계이다"

$$\text{Occupancy} = \frac{\text{(응대시간 + 마무리 시간)}}{\text{(응대시간 + 마무리시간 + 대기시간)}}$$

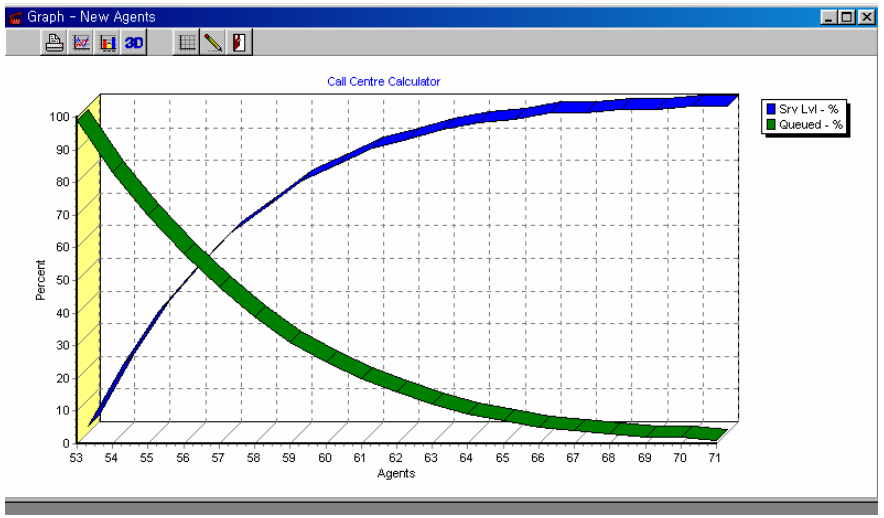
- Occupancy가 올라가면 Service Level은 내려감
- 상담원 그룹의 크기도 Occupancy에 영향을 끼침
  - 상담원 그룹이 커질수록 같은 서비스 레벨 하에서 Occupancy는 커짐
- Occupancy는 직접 조절할 수 없음
  - 상담원들은 "Occupancy"가 아닌 "adherence factor"에 신경 써야 함

The Laws of Call Center Nature *cont'd*

- The Law Of Diminishing Returns

Agents	서비스 레벨	ASA (Average Speed of Answer)	Occupancy
1	26%	171	97%
2	46%	68	95%
3	61%	36	92%
4	72%	21	90%

"상담원 한명이 추가로 투입될 때마다, 증가된 상담원으로 인한 service level의 개선되는 정도는 감소할 것이다."



Anton Consulting

The Laws of Call Center Nature *cont'd*

The Powerful Pooling Principle

"자원의 통합은 통화량의 효율성을 증대시킨다."

EX)

처리 콜	서비스 레벨	필요 상담원	Occupancy
25	80% in 20 sec.	5	58%
50	80% in 20 sec.	9	65%
100	80% in 20 sec.	15	78%
500	80% in 20 sec.	65	90%

15명으로 구성된 하나의 그룹이 9명으로 구성된 2개의 그룹과 동일한 서비스 레벨로 동일한 콜 로드를 처리할 수 있다.

The Laws of Call Center Nature *cont'd*

- The Powerful Pooling Principle (cont'd)
  - 효과
    - 동일한 수의 상담원으로 동일한 서비스 수준에서 더 많은 콜 처리
    - 보다 적은 수의 상담원으로 동일한 서비스 수준에서 동일한 수의 콜 처리
    - 동일한 수의 상담원으로 동일한 수의 콜을 더 좋은 서비스 레벨에서 처리
  - pooling vs. specialization
    - pooling과 specialization의 연속선 상에서 생각해야 함.
    - 일반적으로 콜센터의 자원이 통합되면 효율성이 증가하지만, 비용 등의 문제를 고려할 때 specialization이 바람직한 경우가 있음 (ex. 서로 다른 언어 또는 뚜렷하게 다른 제품군).
    - 이러한 경우를 제외하고, 통합성을 높이기 위해 cross-training의 기회를 (모든 상담원이 어떤 call이라도 처리할 수 있도록) 모색해야 함