ABL Digital Technologies

MULTI-SKILL SIZING

A Practitioner's Guide

The pooling-leverage maths · Every input explained
The blended sizing chain · The Excel workbook · A worked example

Companion to the Erlang and Shift Planning explainers.

MULTI-SKILL SIZING

A Practitioner's Guide

The pooling-leverage maths · Every input explained
The blended sizing chain · The Excel workbook · A worked example

1 · What This Tool Does

The Multi-Skill Sizing tool answers one question: when several contact-centre queues are staffed by a partially-shared pool of agents, how many people do you really need — and how much do you save versus rigidly separate teams?

It does four things, in order:

- Sizes each queue independently with Erlang C — the "no pooling" baseline
- Sizes the combined queue (volume-weighted AHT and SLA) — the "full pooling" baseline
- Interpolates between the two using a Leverage Factor you control — the realistic operating point
- Reports the savings, the per-queue occupancy picture, and a sensitivity curve

> *For the underlying Erlang C maths, see the Erlang Formulas Explainer. For single-queue staffing in detail, see the Shift Planning Explainer.*

2 · The Maths At A Glance

2.1 Independent Per-Queue Sizing

For each queue with its own calls/hour, AHT and SLA target, the Erlang C requirement is computed independently. A min-staff floor is applied where volume is positive.

```
Required[q] = max(MinStaff, agentsRequired(SLA[q], SLATime[q], Calls[q], AHT[q]))

SumIndependent = Σ Required[q]
```

2.2 Full-Pooling Baseline

Combine all queues into a single virtual queue with volume-weighted parameters:

```
TotalCalls = Σ Calls[q]

BlendedAHT = Σ (Calls[q] × AHT[q]) ÷ TotalCalls

BlendedSLATarget = Σ (Calls[q] × SLA[q]) ÷ TotalCalls
```

```
BlendedSLATime = Σ (Calls[q] × SLATime[q]) ÷ TotalCalls
```

```
PooledRequired = agentsRequired(BlendedSLATarget, BlendedSLATime,
TotalCalls, BlendedAHT)
```

PooledRequired is the lowest possible required headcount — the best case when every agent can answer every queue and routing is perfect.

2.3 Blended (Realistic) Sizing

Real operations sit between the two extremes. The Leverage Factor — a value between 0 and 100% — controls how close to full pooling the practical operation can get:

```
Blended = ■ SumIndependent × (1 − leverage) + PooledRequired ×
leverage ■
```

Typical practical leverage:

- 0% — rigid skill separation, no cross-training (rarely seen in modern operations)
- 30–50% — partial multi-skill, one agent covers 1.5–2 queue types on average
- 50–70% — generalist agents with priority routing
- 80–100% — universal agents (theoretically possible, operationally rare)

2.4 Roster FTE & Savings

```
Roster FTE = ■ Blended ÷ (1 − Shrinkage) ■
```

```
Leverage Savings = SumIndependent − Blended (in agents)
```

```
Leverage Savings % = Savings ÷ SumIndependent
```

3 · The Inputs

3.1 Queue Configuration

2 to 6 queues. Add or remove with the buttons above the table. Each queue has:

- Queue Name — for the report (Sales / Service / Collections etc.)
- Calls / Hour — the volume rate for this queue
- AHT (sec) — average handle time for this queue
- SLA Target (%) — service level target
- SLA Time (sec) — wait threshold for the SLA

Each queue can have its own AHT and SLA target. A high-touch queue (long AHT, looser SLA) and a transactional queue (short AHT, tight SLA) often share the same pool of agents.

3.2 Shared Parameters

Shrinkage (%)

Used to inflate the blended in-seat requirement into rostered FTE. Standard 22–32%.

Occupancy Cap (%)
Informational only — shown in the per-queue table. Does NOT inflate Required (consistent with the rest of the tool set).

Min Staff per Queue
A floor for each queue independently. Useful for ensuring at least one agent is dedicated even when Erlang C would return zero.

Leverage Factor (%)
The key knob. Pick from the typical bands in section 2.3 above. The sensitivity chart shows what happens across the whole 0–100% range — useful for understanding how much the answer depends on this assumption.

4 · The Outputs

4.1 Sizing Summary Card

- Sum independent (no pooling)
- Full pooling (best case)
- Blended at chosen leverage
- Roster FTE after shrinkage
- Leverage savings in agents and %
- Total load in Erlangs
- Blended occupancy

4.2 Per-Queue Breakdown Table

Each queue's calls, AHT, load, independent requirement, occupancy and achieved SLA at independent staffing. Occupancy is colour-coded — red if over the cap, gold within 85% of the cap, green otherwise.

4.3 Sensitivity Chart

Required and Roster across the full 0–100% leverage range, with the currently selected leverage marked. If the curve is steep, your answer is sensitive to the leverage assumption — be conservative. If it is flat, the assumption matters less.

5 · The Excel Workbook — Four Sheets

5.1 Summary

Shared parameters, aggregate sizing, and headline savings.

5.2 Per-Queue

Every queue with its own load, requirement, occupancy and achieved SLA.

5.3 Leverage Sensitivity

Required headcount and Roster FTE across 0–100% leverage in 10% steps, with the savings vs independent at each level.

5.4 Assumptions

Methodology narrative — the four steps of the maths and what the tool does NOT cover (skill-overlap matrices, priority routing, hard skill walls, cost-per-skill differences).

6 · A Worked Example

Three queues sharing a common agent pool. The defaults the tool loads with:

- Sales — 200 calls/hour, 240 sec AHT, 85%/20s SLA
- Service — 350 calls/hour, 320 sec AHT, 80%/30s SLA
- Collections — 100 calls/hour, 400 sec AHT, 75%/30s SLA

Shrinkage 25%, Leverage 50%.

6.1 Per-Queue Independent

```
Sales: Load = 200×240/3600 = 13.3 E → Required ≈ 18 agents
Service: Load = 350×320/3600 = 31.1 E → Required ≈ 36 agents
Collections: Load = 100×400/3600 = 11.1 E → Required ≈ 15 agents
SumIndependent ≈ 69 agents
```

6.2 Full Pooling

```
TotalCalls = 650
BlendedAHT = (200×240 + 350×320 + 100×400) ÷ 650 ≈ 305 sec
BlendedSLA = (200×0.85 + 350×0.80 + 100×0.75) ÷ 650 ≈ 81%
BlendedTime ≈ 26 sec
PooledRequired ≈ 60 agents
```

6.3 Blended at 50% Leverage

```
Blended = ■ 69 × 0.5 + 60 × 0.5 ■ = ■ 64.5 ■ = 65 agents
Roster = ■ 65 ÷ 0.75 ■ = 87 FTE
Savings = 69 − 65 = 4 agents (~6%)
```

6.4 Reading the Result

Moving from rigid skill separation to 50% leverage saves you 4 in-seat agents (~6 rostered FTE). The sensitivity chart will show that pushing to 70% leverage saves a few more; pushing to 100% would save 9 agents but is rarely operationally feasible. Use 50% as the planning number, document that the tool says 70% saves another 4 agents if cross-training improves.

7 · Limits

The tool deliberately does NOT model:

- Skill-overlap matrices — specific maps of which agent groups can take which queues. The linear leverage interpolation is a simplification.
- Priority routing — when high-priority calls displace lower-priority ones. The pooled baseline assumes neutral routing.
- Hard skill walls — agents who categorically cannot take a given queue (language, certification). For these, exclude them from the pool and size separately.
- Cost-per-skill differences — the tool returns headcount, not cost-weighted choices.

For any of these, hand the numbers to a multi-skill discrete-event simulator — Erlang-style closed-form maths cannot answer them precisely. But for a first-cut commercial estimate, this tool gives you a defensible range bracketed by the 0% and 100% leverage extremes.

8 · Tips

How do I pick the leverage factor?

Empirically: track the fraction of calls in your operation that were actually answered by a "non-primary" agent. If 40% of your sales calls go to a service agent during peaks, leverage is at least 40%. If you have no data, use 50% as a defensible middle.

Why does the SLA achieved at independent staffing look so high?

Because independent staffing is the conservative answer — it gives each queue exactly the agents needed for its target SLA, with a small buffer. Once you blend down to a lower headcount, achieved SLA falls toward the target.

Why is pooling savings smaller than I expected?

The Halfin-Whitt √N rule: pooling savings grow with the square root of load, not linearly. A 10 Erlang pool of two 5 Erlang queues saves around 15-25% of agents, not 50%. Bigger queues pool better than small ones.

Where can I learn the underlying queueing theory?

Hyndman & Athanasopoulos for forecasting; Halfin & Whitt 1981 for the heavy-traffic pooling result; Koole and Mandelbaum for multi-skill queueing. The Erlang Formulas Explainer covers the single-queue maths in plain language.