LTVplus

STOP COPYING OTHER COMPANIES' SCALING PLANS.... BUILD YOUR OWN!

How to Build a Scaling Checklist That Actually Works for Your MSP

Everyone wants a 10-step checklist to scale support operations.

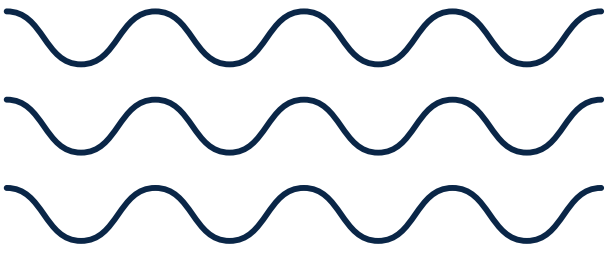
They search for a formula, find something that worked for another company, and try to copy-paste it into their own business. Sometimes it works. More often, it doesn't.

BUT WHY?

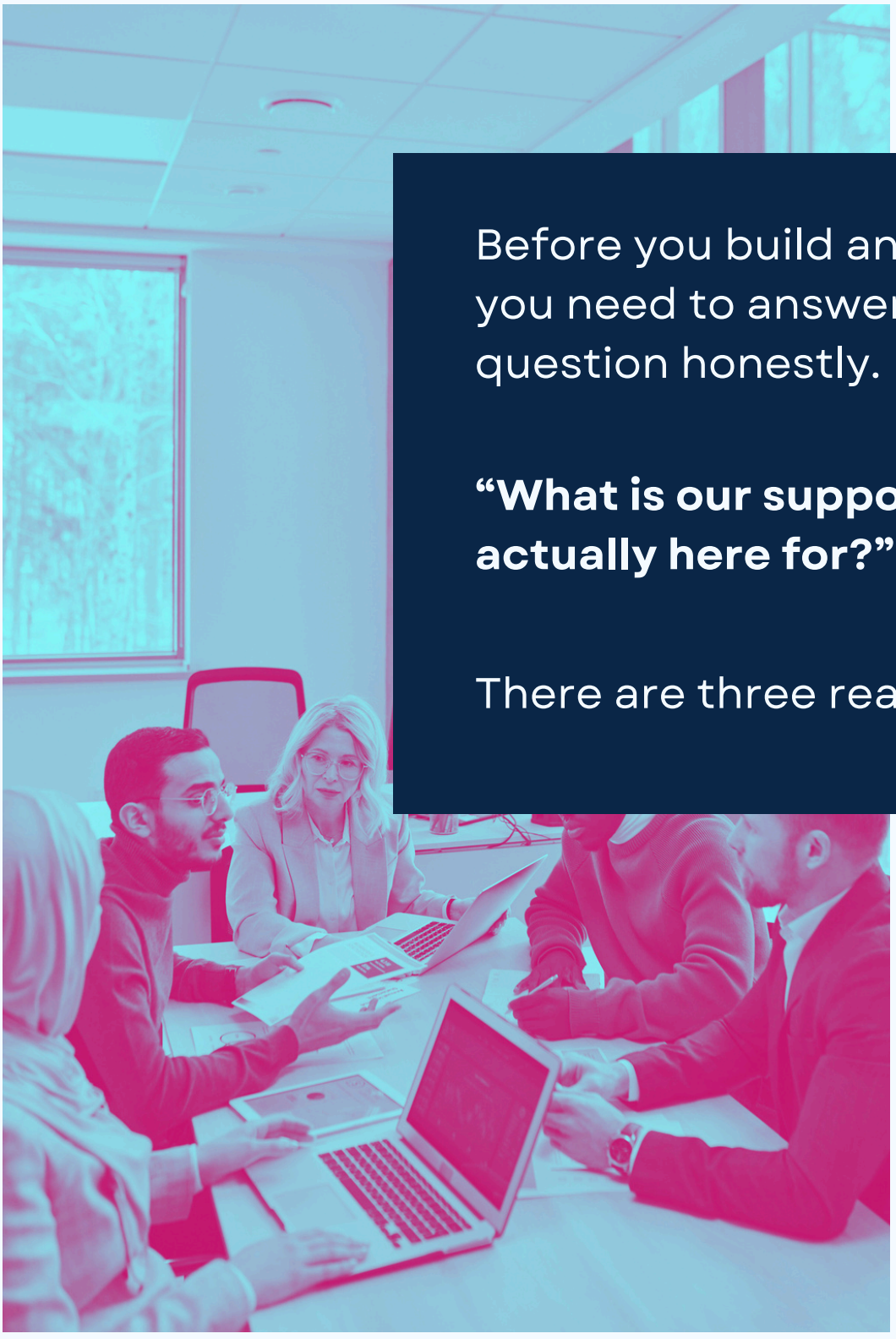
Because checklists are tactics. And tactics without strategy are just a list of things to do

The MSPs that scale well.. They're building their own plan, based on their clients, their team, and where their operation actually is right now.

THIS GUIDE WILL HELP YOU DO EXACTLY THAT.



FIRST: DEFINE YOUR SUPPORT PHILOSOPHY



Before you build any checklist, you need to answer one question honestly.

“What is our support function actually here for?”

There are three real answers:

Cost Center:
Support exists to resolve issues efficiently. Speed and cost-per-ticket are the metrics that matter. This works if your client relationships are managed at the account level, not the helpdesk. Support built for fast response and maintaining SLA’s

Profit Center:
Support drives retention and expansion. Engineers are trained to spot upsell signals. CSAT and Net Revenue Retention are what you're measuring.

Intelligence Lab:
Support surfaces risks and opportunities across your client environments. Ticket data feeds QBRs, information security practices, and your proactive recommendations.

Most MSPs say they're a strategic partner but operate like a cost center. That gap is where client trust quietly erodes.

Get clear on which one you actually are. Everything else builds from here.

THE 10-STEP FRAMEWORK

The questions at the end of each step are there to help you figure out where you actually stand, not where you think you stand.

Step 1:
Audit What You're Actually Delivering

Before you scale anything, you need an honest picture of current delivery. Not what your SLA document says but what's actually happening.

Pull 90 days of ticket data and look at:

- What percentage of issues are resolved at Tier 1 versus escalated?
- What's your actual response and resolution time against contracted SLAs?
- Which client environments are generating the most noise, and why?

You need to measure your current state before you design your future state, it helps you see where time and effort are really going.

You can't fix what you haven't honestly looked at.

Ask yourself: Where is the real bottleneck in your delivery chain right now? Is it a people problem, a process problem, or a tooling problem?

Step 2:
Define the Tiers You're Actually Staffing For

Most MSPs have a Tier 1/2/3 structure that looks clean on paper and is messy in practice.

- Engineers get pulled up because Tier 1 isn't trained deeply enough
- Senior engineers spend time on L1 tickets because escalation criteria aren't clear
- Everyone's covering the gaps, but nobody's saying it out loud

Having frameworks that define clear ownership and handoff points across tiers, and across any vendor or external teams you're working with. If you're using outsourced capacity, this clarity becomes even more critical.

If your escalation structure isn't explicit, it doesn't exist!

Ask yourself: If your Tier 1 team had to operate for four hours without escalation access, what would break? What does that tell you about your knowledge transfer gaps?

Step 3: Standardize Before You Scale

Knowledge-Centered Service developed by the Consortium for Service Innovation, is one of the most validated frameworks for scaling support.

The principle:
knowledge gets created and captured as a byproduct of solving problems, not as a separate documentation project.

For MSPs, this means engineers are building and refining runbooks and resolution guides every time they close a ticket. Not in quarterly documentation sprints. Every single ticket.

If your team can't resolve your most common issues from a documented process, you're scaling your knowledge gaps alongside your headcount.

Ask yourself: What are your top 10 ticket categories by volume? Does a documented resolution process exist for each one?

Step 4:
Build a Coverage Model That Reflects Reality

Most MSPs underestimate coverage complexity until it's already causing delivery problems. Especially if you've:

- Added clients across different time zones
- Taken on contracts with 24/7 SLA commitments
- Recently absorbed another firm's client base

With demand forecasting: when are tickets actually coming in, at what volume, and what skills do they require? You build staffing patterns around real data, not around what's always been done.

This is also where structured external capacity starts to make operational sense. When demand is high-variance or you need around-the-clock coverage without burning your internal team, a reliable overflow model isn't just a cost decision.

“Staffing for the business you had last year will fail the business you have today.”

Ask yourself: Do your current staffing patterns reflect actual ticket demand, or are they based on assumptions your client base has already outgrown?

Step 5:
Define What "Good" Looks Like at Every Level

Scaling breaks down fast when there's no shared definition of good performance.

- Engineers don't know what they're actually being measured on
- Managers make inconsistent calls
- New hires get evaluated on gut feel

The fix is building a simple competency framework for each role in your support operation. Here's how to do it:

1. Map your roles to responsibility levels.
2. Define the skills each role requires.
3. Write performance expectations, not just job descriptions.
4. Make career progression visible.

Unclear expectations are a retention problem disguised as a performance problem.

Ask yourself: Could your engineers clearly explain what they need to demonstrate to move to the next tier? Would their manager give the same answer?

Step 6:
Onboard Clients the Way You Onboard Staff

This is where most MSP growth stories develop their first cracks.

The deal gets closed. Ops gets handed a new environment, partial documentation, and a go-live date. And the existing client base starts to feel the distraction.

Moving a new client from signed to live without destabilizing what's already running. That means:

- Defined onboarding stages with clear internal ownership
- Documentation requirements before go-live
- A hypercare period with explicit exit criteria

Every chaotic client onboarding is a process gap, not a people problem.

Ask yourself: Do you have a written onboarding protocol your ops team follows consistently, or does every new client look different depending on who's leading it?

Step 7:
Build an SLA Structure That Matches Service Complexity

As your client base grows, a flat SLA structure becomes a liability. Clients with different environments, risk profiles, and contract values shouldn't be on identical commitments.

A growing number of MSPs are also layering XLAs (Experience Level Aggreements) alongside traditional SLAs.

- An “SLA” measures what you did (response time, uptime percentage)
- An “XLA” measures how the client experienced it

That's a meaningful distinction if you're positioning as a strategic partner rather than a transactional helpdesk.

Hitting your SLA numbers while losing client trust is a sign your metrics aren't measuring the right things.

Ask yourself: Are your current SLAs creating the right incentives for your team, or are engineers technically hitting targets in ways that don't reflect actual client experience?

Step 8:

Build a Major Incident Process That Doesn't Live in Someone's Head

Every MSP has a war story. A Priority 1 incident that went sideways because no one knew who was in charge. Communication that broke down. The right people pulled in too late.

ICS (Incident Command System), originally developed for emergency response and adapted into IT major incident management, solves this by defining clear command roles, communication protocols, and decision rights before anything goes wrong.

When a client environment goes down at 2am, your team executes a defined process. They don't figure it out in real time.

Clients tolerate outages. What they don't tolerate is silence.

Ask yourself: If a P1 happened tonight, would every person involved know their specific role? Would your client get a proactive update within the first 30 minutes?

Step 9:
Turn Support Data Into Business Intelligence

Your support team generates valuable signals every single day. Engineers are seeing:

- What's breaking and why
- What clients are confused or frustrated by
- What risks are quietly emerging across environments

If that intelligence isn't feeding your account management, QBRs, Information Security Officer, or tooling decisions, it's sitting in your ticketing system doing nothing.

A simple, structured debrief between support leadership and account leadership (weekly or monthly, depending on your size) can turn your helpdesk into one of your most valuable business intelligence assets.

Most MSPs are sitting on a goldmine of client insight and calling it a ticket queue.

Ask yourself: In the last quarter, how many client retention or expansion decisions were directly informed by patterns your support team surfaced?

Step 10: Design for Absorbing Growth, Not Just Current State

This is the step that determines whether you can actually grow without things wobbling.

Building a resourcing model that flexes with demand means two things:

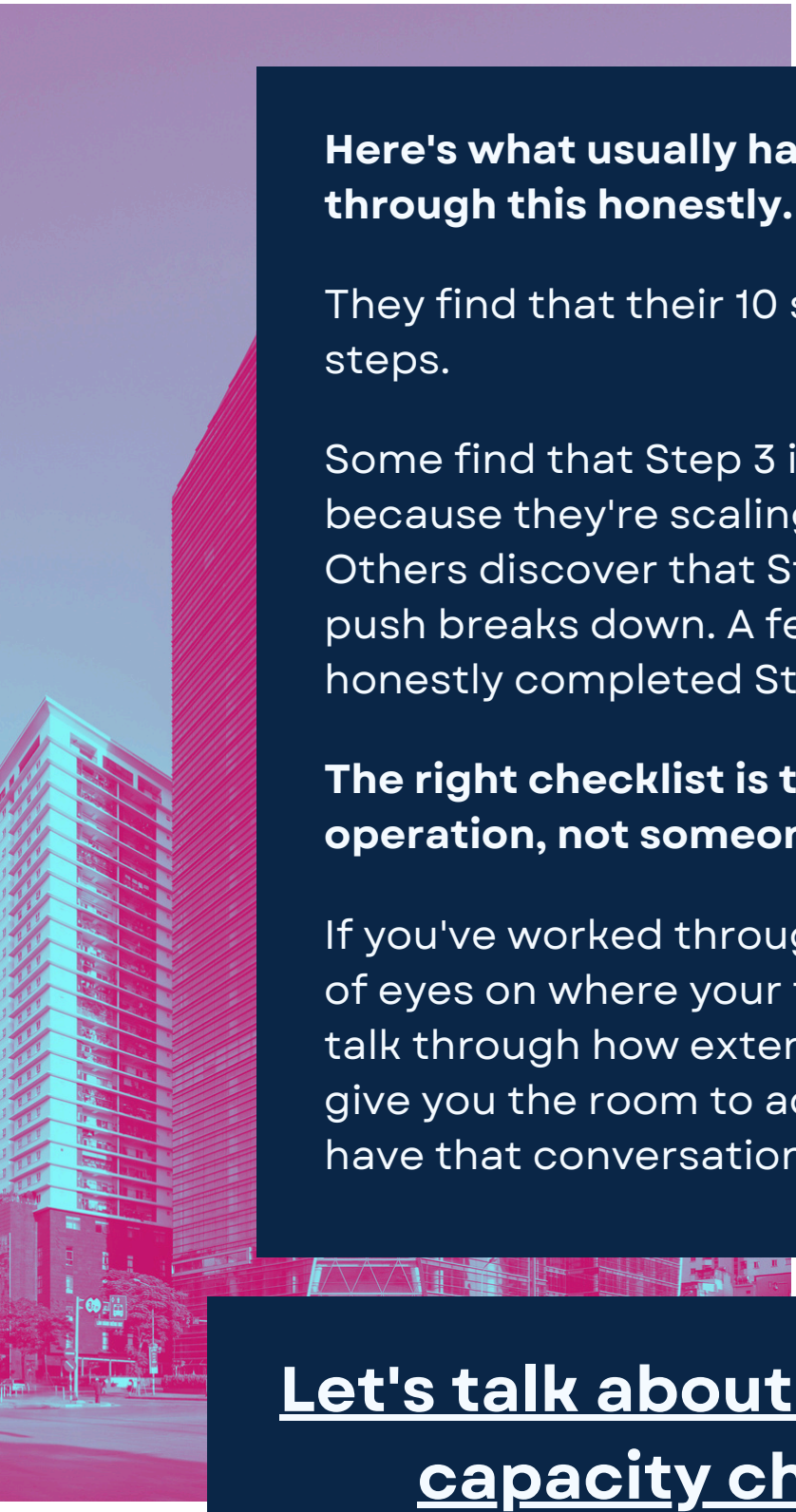
- **Knowing your internal capacity threshold**: the point at which adding workload starts degrading delivery quality
- **Having a clear, written plan for when you hit it**: cross-training programs, overflow protocols, or a structured relationship with an external workforce partner

The goal isn't idle capacity sitting on the bench. It's a fast, predictable path to additional support when demand spikes, whether that's absorbing a new acquisition, covering an unexpected departure, or meeting a client who just signed a more demanding SLA.

The MSPs that absorb growth well aren't just good operators. They planned for it.

Ask yourself: Do you know your current internal capacity threshold? If a major new contract came in next week, what would you actually do?

SO WHAT'S YOUR CHECKLIST?



Here's what usually happens when leaders work through this honestly.

They find that their 10 steps don't look like these 10 steps.

Some find that Step 3 is where everything stalls because they're scaling on undocumented processes. Others discover that Step 6 is where every growth push breaks down. A few realize they've never honestly completed Step 1 with real data.

The right checklist is the one built around your operation, not someone else's.

If you've worked through this and want a second set of eyes on where your team stands, or you want to talk through how external support capacity could give you the room to actually execute, we're happy to have that conversation.

Let's talk about your current capacity challenges