

THE COMPANION PAPER

The Outreach Engine

A rotation scheduler, a daily cap, a carry forward rule, and a ranked morning list, for anyone who runs outreach at scale.

Companion to the essay "102 Percent," michaelgaff.com/journal/102-percent
Architecture and wiring only. No donor data, no credentials, no private endpoints.

Michael Gaff · michaelgaff.com

1. System overview and scope

The outreach engine is a single service that decides, every day, who should be contacted, by what channel, and in what order. It does not decide what to say. It does not hold the relationship. It removes the bookkeeping that used to sit between having hundreds of relationships and actually tending to them on schedule.

Scope, on purpose, is narrow. The engine owns four jobs: rotation, capping, carry forward, and ranking. Everything else, the actual conversation, the judgment about tone and timing, the decision to escalate or hold back, stays with the person running it.

2. The rotation and cap logic

Every contact in the system carries a cadence: how often they should be touched, and by which channels are appropriate for them. The engine walks the full list nightly and computes who is due the next day based on the last logged touch date plus their cadence interval.

A hard daily cap sits on top of that list. The cap exists because an uncapped queue on a slow week silently turns into an avalanche on a busy one, and an avalanche is what gets skipped entirely. The cap is configurable per channel, for example a ceiling on calls separate from a ceiling on emails, because the effort per channel is not equal.

- Cadence: minimum and maximum days between touches, set per relationship, not global.
- Channel mix: calls, emails, and letters each carry their own daily ceiling.
- Open day: one weekday intentionally carries no cap, reserved for catch up rather than new outreach, so a light week never gets mistaken for a broken scheduler.

3. The carry forward rule, and why it matters

If a due contact is not marked touched by the end of the day, the engine does not drop them. It re queues them at the front of the following day's list, ahead of that day's new due contacts, with a small escalating priority weight the longer they wait.

This single rule is the difference between a scheduler and a nag. Without carry forward, a missed day quietly becomes a missed relationship, because tomorrow's list looks identical to any other day. With it, nothing gets permanently lost to a bad Tuesday.

4. The ranked daily output

Each morning the engine emits one artifact: a short, ordered list of exactly who to reach that day and by what means. Ranking blends three signals: how overdue the contact is, a manually settable priority tier for especially important relationships, and a light recency boost for anyone recently engaged, so

momentum is not accidentally interrupted.

The output is intentionally plain. A ranked list, not a dashboard. The goal is a document that takes ten seconds to read and tells a person exactly where to start if the day gets cut short.

5. The sync layer and data freshness

The engine reads from, and writes touch logs back to, the system of record on a nightly synchronization job. It never treats its own queue as authoritative. If a touch was logged directly in the system of record outside the engine, the next sync reconciles it and the contact drops off the due list automatically.

Freshness matters more than cleverness here. A same day stale read is worse than a simple algorithm on current data, so the design favors a nightly full resync over a more complex incremental one, trading a small amount of compute for a much smaller number of edge cases.

6. Failure modes and the human override

If the sync fails, the engine serves the last known good list rather than an empty one, and flags it as stale so the person running it knows to double check before relying on it. If the cadence data for a contact is missing or malformed, that contact is surfaced in a separate exceptions list rather than silently dropped or silently over contacted.

Nothing in the engine sends anything on its own. It produces a list. A person decides whether, how, and when to act on each line of it. That boundary is not a limitation of the current build. It is the design, and it is the reason the tool is trustworthy enough to run every single morning without supervision.

7. Design principles

- Boring on purpose. No machine learning, no black box ranking, just a small number of explainable rules anyone can audit in a sitting.
- The cap protects the person, not the process. Better to under queue than to burn someone out with an unworkable list.
- Carry forward over guilt. The system compensates for a missed day instead of just noting the failure.
- The list is a tool, not a taskmaster. It informs judgment. It never replaces it.

Companion to “102 Percent” on michaelgaff.com. Built by one human and one AI partner. No agency, no staff.