A Function Paper from GRAIL

What does an AI-augmented IT department look like?

IT department

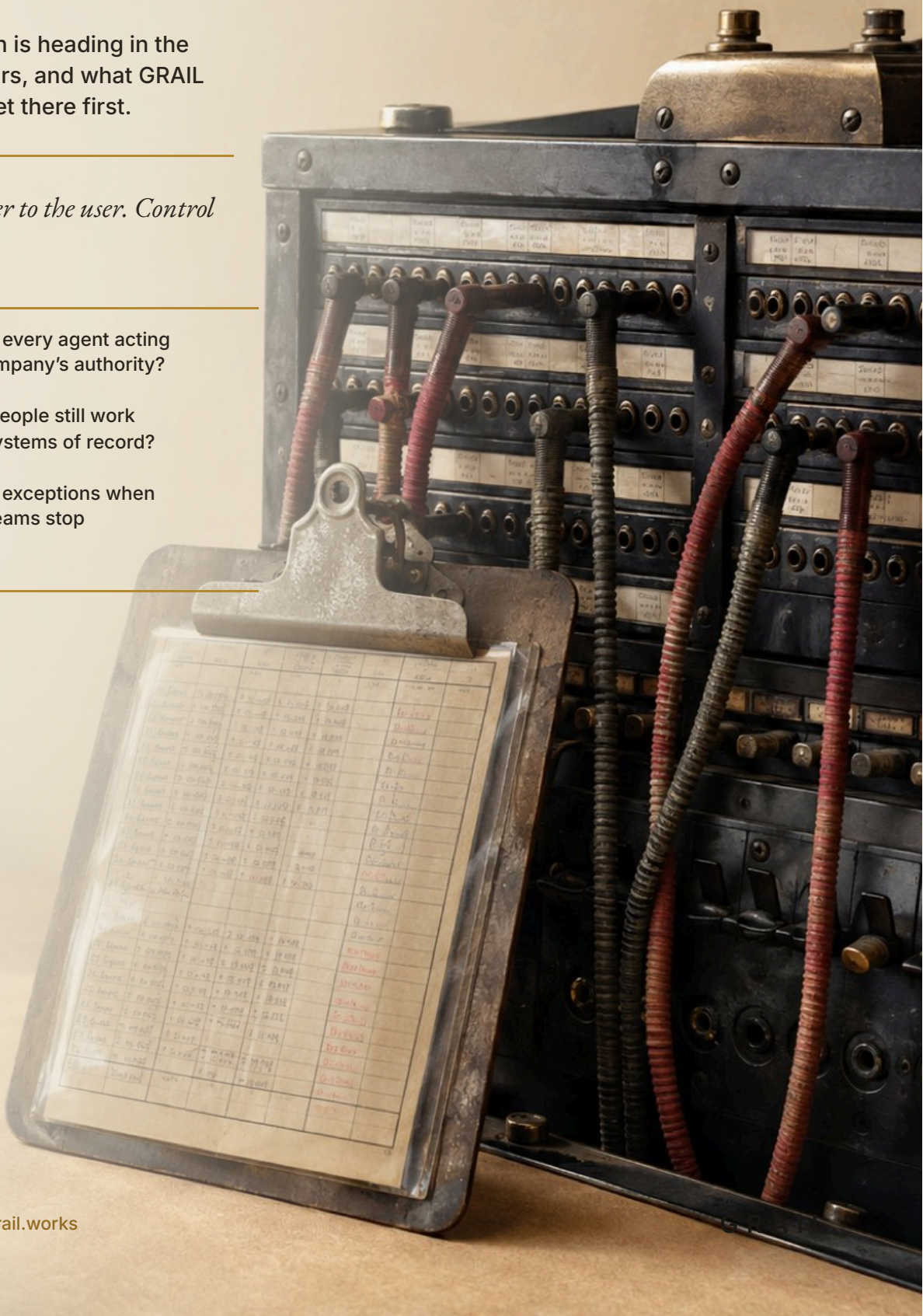
Where the IT function is heading in the next two to three years, and what GRAIL believes it takes to get there first.

"The work moves closer to the user. Control stays with IT."

01 Can you name every agent acting under your company's authority?

02 Why do your people still work around your systems of record?

03 Who owns the exceptions when your service teams stop processing?



More insights like this at grail.works

WHAT DOES AN AI-AUGMENTED IT DEPARTMENT LOOK LIKE?

Where the IT function is heading in the next two to three years, and what GRAIL believes it takes to get there first.

IT has spent decades receiving requests, joining systems and protecting access one project at a time. **The structural change is that IT becomes the steward of how every company agent reads, reasons and acts across those systems.** The business gains reusable capabilities. IT sets the boundaries.

THE IT FUNCTION IN 2028 AS A CONTROLLED FLOW

01 EMPLOYEE OR AGENT

Identifies the employee, device, entitlements and recent changes.

02 APPROVED CAPABILITY

Retrieves a permission-aware answer or proposes an approved action.

03 IDENTITY AND PERMISSION

Checks identity, permissions and branch controls before action.

04 SYSTEM OF RECORD

Keeps every interaction as a ticket in authoritative records.

05 AUDIT AND ROLLBACK

Retains each accountable person, approval and rollback plan.

The work moves closer to the user. Control stays with IT.

GRAIL's view of the operating model now coming into reach. A position, not a survey.

THE FUNCTION TODAY

The queue sits between the company and its *systems*

The IT function usually combines a CIO or CTO, service-desk staff, infrastructure and cloud engineers, security specialists, application owners, enterprise architects, integration developers, project managers and vendor managers. In smaller companies, one person may carry several of those roles. Managed-service providers often sit inside the operating model.

The function keeps the company running. It shapes uptime, operating cost and risk. It also shapes the top line when customer integrations, proprietary software or delivery capacity determine how quickly the company can sell and serve customers.

Eight processes

01	Service desk and end-user support	05	Integration, data plumbing and system interfaces
02	Infrastructure, cloud and workplace operations	06	Software development and maintenance
03	Security, identity, access and technical compliance	07	Business technology projects and change delivery
04	Application portfolio, vendors, licenses and technology cost	08	Architecture, capacity planning and technology governance

The information required to run them is scattered. Tickets sit in ServiceNow or Jira Service Management. Identity and endpoint platforms know the user and device. Cloud consoles, monitoring systems and the SIEM know what changed. Git repositories and delivery pipelines know what shipped. Project tools, ERP, CRM, vendor portals, spreadsheets and architecture diagrams hold different parts of the same story.

The systems contain the events. People still carry too much of the explanation.

The reasoning behind exceptions, integrations and past technology decisions often remains in specialists' heads. That makes every incident, renewal and project depend on finding the right person before the company can act.

WHAT CHANGES STRUCTURALLY

IT becomes the access and *control* function

The function is being reshaped along two fundamental lines.

IT begins to run an internal product for agent access. Every department needs governed ways for its agents to read and act in ERP, CRM, finance, HR, ticketing and document systems. IT owns the connector layer, agent identities, permission model, approval rules, tool registry, audit records and emergency controls. A connection is no longer built once for one request. It becomes a reusable business capability such as reading customer status, creating a support case or preparing a purchase order.

Operations move from receiving requests toward continuous sensing and prevention. Service patterns, infrastructure drift, access anomalies, unused licenses, delivery risks and repeated integration failures are found before somebody opens a ticket. Software engineering follows the same shift. Producing routine code becomes easier. Specifying the right behaviour, reviewing changes and protecting system integrity become more valuable.

Business teams stop waiting for every simple report or system action to pass through an integration queue. They use approved capabilities while IT controls identity, permissions, records and consequences. The function is measured partly by the safe business capacity it enables, not only by uptime, tickets and budget.

> IT stops being the department that operates every screen. It becomes the steward of who and what may call the process underneath it.

WHERE IT IS WEAKEST Complex legacy changes, production remediation and cross-system exceptions remain difficult. More generated output can add review work when architecture, knowledge and quality controls have not changed with it.

THE 2028 OPERATING PICTURE

A day, a week, a *month*

A heavily invested IT function in 2028 runs through a briefing and approval layer over its systems of record. Employees and technicians work through Teams, Slack, an IDE, the company portal or a daily briefing. The underlying systems keep the authoritative records.

The day

MORNING

The operations team receives one prioritised view of service degradation, security events, failed integrations, cost anomalies, software delivery risks and decisions awaiting approval. Agents have correlated tickets, telemetry, recent deployments, asset records and earlier incidents. Staff investigate the cases where evidence conflicts, consequences are material or no approved playbook applies.

DURING THE WORK

Routine support begins in ordinary language. The workflow identifies the employee, device, entitlements and recent changes. It retrieves a permission-aware answer or proposes an approved action. Infrastructure staff receive a likely cause, affected services and a rollback plan rather than a stream of disconnected alerts. Engineers give coding agents bounded migrations, tests, documentation, dependency changes, defect investigations and well-specified features.

BEFORE ACTION

People approve exceptional access, consequential production changes and unclear support cases. Engineers own acceptance criteria, architecture and behaviour. Branch controls determine what may merge. Every interaction remains a ticket, and every material action retains an accountable person.

THE WEEK The function reviews exceptions and recurring failures. Service specialists improve knowledge and self-service journeys. Platform staff turn repeated integration work into governed tools. Security staff update policies and detection playbooks. Application owners compare usage, cost and business dependency before renewal conversations.

THE MONTH The CIO or CTO reviews operational resilience, agent access, engineering capacity and technology economics. The discussion covers permissions added, actions denied, human overrides, incidents to which agents contributed and business requests that should become reusable capabilities.

Continuous diagnosis, permanent access control and reusable system capabilities used to be separate ambitions. They become one operating layer, with human judgment concentrated where consequences are real.

PROCESS BY PROCESS

What runs, and what stays with the *person*

PROCESS	WHAT THE AGENT DOES	WHAT STAYS WITH THE PERSON
Service desk and support	Structures the issue, retrieves approved knowledge, drafts replies, verifies the user and device, and proposes standard resets or software access	Ambiguous cases, emotional situations, exceptions and approval of consequential device or access changes
Infrastructure and cloud operations	Correlates alerts, configuration, deployments and tickets; prepares an incident narrative, likely causes and a tested remediation or rollback plan	Diagnosis where evidence conflicts; approval of production changes with meaningful blast radius
Security, identity, access and compliance	Correlates alerts across identity, endpoint, email, vulnerabilities and the SIEM; explains suspicious activity and drafts response steps	Containment, privilege and exception decisions; policy and detection design
Applications, vendors, licenses and cost	Assembles usage, duplication, contract terms, invoices, support history, renewal dates and business dependency	Negotiation strategy, service-risk decisions, acceptable lock-in and approval to remove access
Integration and data plumbing	Proposes mappings, generates adapters, tests schemas and exposes versioned business capabilities through a shared gateway	Semantics, permissions, ownership, failure behaviour and approval of writes
Software development and maintenance	Takes bounded changes, creates tests and documentation, investigates defects, proposes migrations and returns a pull request with provenance and risk	Specification, architecture, review, production acceptance and hard failures
Business technology projects	Turns decisions into requirements, dependencies, actions and risk updates; keeps work items and schedules current	Scope conflicts, supplier choices, trade-offs and priority
Architecture and technology governance	Maintains dependency maps, finds policy exceptions and technical-debt patterns, and proposes standards from current system evidence	Target architecture, acceptable exceptions and technology capital allocation

The right-hand column is the function's future centre of gravity. The agent assembles, correlates and proposes. The person decides what the company may depend on.

DATA AND CONNECTIONS

Four stages on *The Access Ladder*

The connector landscape is usable but uneven. The path should follow business consequence, not technical novelty. These four stages map the source's connection sequence onto the rungs of GRAIL's Access Ladder.

01 **Identity, collaboration, knowledge and ITSM** ACCESS LADDER: DOCUMENTS ONLY, THEN READ ACCESS

Begin with knowledge articles, policies, runbooks, architecture records, contracts and past incident reports. Add permission-aware identity, collaboration and ITSM reads to establish user context, entitlements and the support record. Outputs remain answers, briefs and proposed tickets until the sources are trusted.

02 **Monitoring, endpoints, security, code and CI/CD** ACCESS LADDER: READ ACCESS, THEN BOUNDED AUTONOMOUS ACTION

Connect the evidence needed for live diagnosis before granting consequential writes. Reversible, predefined actions such as an approved reset, restart or rollback may run inside defined limits. Staff approve actions with material consequences.

03 **ERP and operational systems** ACCESS LADDER: READ AND ACT WITH APPROVAL

These connections expose financial and operational records. Production writes, access changes and vendor commitments require a named approver. Destructive changes, privileged access and actions separated by duty retain dual approval or the existing change board.

04 **A lake or warehouse** ACCESS LADDER: THE DATA FOUNDATION ACROSS EVERY RUNG

Point connections can retrieve one ticket or perform one approved action. Cross-system questions need common identifiers, history and quality rules. The data layer shows which deployments increase tickets, which applications cost more than their business value and which access patterns predict incidents.

The connector landscape in September 2026

Microsoft Work IQ	Permission-aware mail, calendars, files, people, chats and sites through governed tools. Its controls cover reads, creation, updates, deletion and actions.
ServiceNow	Selected APIs, actions, playbooks and knowledge resources can be exposed while existing roles and access controls continue to apply.
Atlassian Rovo	Governed reads, creation and updates across Jira, Confluence, Compass, Jira Service Management and Bitbucket, with inherited permissions and audit logs.
GitHub	A remote server with OAuth, configurable read-only modes, tool allowlists and enterprise policy controls.
Business Central	Read-only by default. Administrators configure the API pages allowed to create, modify, delete or perform bound actions under the user's identity.
Spiris	A hosted service in beta with OAuth and selected read-write accounting tools. Visma Business NXT and Visma Net remain API-first.
Fortnox	REST APIs, OAuth scopes and service-account authorisation are available. No first-party MCP service was found.
SAP Business One	Service Layer exposes OData and HTTP. Its 2026 release includes sample MCP code rather than a managed production service.
IFS Cloud	OData REST projections and OpenAPI specifications are available. No first-party MCP service was found.
Monitor G5	Read-only API access is available. Write access requires an additional license, and no vendor MCP service was found.

> Headless ITSM does not remove ServiceNow or Jira. It keeps the ticket, workflow, SLA and audit record there while employees and technicians work through the surface that fits the moment.

Every transaction should identify the human initiator, agent, model, sources, tool arguments, authorisation scope, policy decision, approver, target result and rollback status. The system of record remains available for exceptions and administration.

GRAIL'S THESIS

Six things we believe about the access *steward*

The technology is only one part of this change. Six operating beliefs determine whether IT becomes a source of safe business capacity or a faster version of the old request queue.

01

The connector layer is a product, not a collection of projects.

A one-off interface solves one request and leaves the next team waiting. A reusable capability has an owner, schema, tests, permissions and failure behaviour. Each later workflow can then use the same controlled route into the company.

02

Identity comes before agency.

An agent that can act without its own identity, scope and expiry is an uncontrolled shared credential. Permission-aware reads establish the starting point. Approval, audit and rollback determine how far action may go.

03

The queue should become a sensor.

Tickets matter, but they describe what somebody has already noticed. The larger prize is continuous attention to degradation, drift, anomalies, unused licenses and recurring failures. Specialists can then spend their time on exceptions, prevention and better playbooks.

04

More code is not the same as more delivery.

Generated changes still consume specification, review, testing and production judgment. Measuring code volume hides review load, rework and instability. The whole flow, through acceptance and operation, is the unit that matters.

05

The system of record should remain authoritative without remaining the main place people work.

Employees can ask for help in Teams. Engineers can act from an IDE. Leaders can decide from a briefing. Tickets, permissions, workflow state and audit history still belong in the systems built to hold them.

06

The role redesign is the investment.

Service specialists become knowledge and exception owners. Administrators move toward platform engineering. Integration developers own connectors and data contracts. Engineers spend more time on behaviour, architecture and hard failures, while leaders decide where the company can safely act faster.

An AI-augmented IT department does not centralise every action inside IT. It gives the company more ways to act while making the boundaries clearer, the records fuller and the consequential decisions unmistakably human.

WHAT IT ASKS OF PEOPLE

Roles, rhythm, and where it *fails*

Service-desk roles move from ticket handling toward knowledge ownership, exception resolution and employee experience. System administrators move toward platform engineering and policy-based operations. Integration developers become owners of connectors and data contracts. Security analysts spend more time on threat hunting, controls and playbook design. Application managers add FinOps and evidence-led vendor management.

Software engineers spend less time producing routine code and more time defining behaviour, reviewing architecture, designing tests and handling hard failures. Junior hiring continues. Juniors pair with senior engineers and agents so routine work changes without removing the learning path that produces the future senior bench.

The team needs workflow design, API and MCP integration, identity engineering, data modelling, evaluation, prompt-injection defence and operational assurance. Deep knowledge of the company's systems remains essential.

The rhythm

TWO TO SIX WEEKS	SIX TO TWELVE MONTHS	18 TO 30 MONTHS	THEN
A document-grounded workflow enters useful work	Teams establish a changed operating rhythm	Reliable action connects several systems	Roles, measures and ownership settle around reusable capabilities

The sequence reflects connector work, control design and behavioural adoption. Model deployment is the smallest part of the operating change.

> WHERE IT FAILS Coding tools arrive without added review capacity. Weak knowledge is connected and repeated at speed. Broad shared credentials replace named identity. Generated volume becomes the measure. The old request queue survives underneath the new interface. A pilot is mistaken for an operating model. Junior roles disappear before a replacement learning path exists.

The function changes when people own the knowledge, permissions, tools and exceptions as permanent work. A pilot can demonstrate the workflow. Only the operating rhythm makes it part of the company.

HAVE YOU THOUGHT ABOUT THIS?

Twelve questions for the IT *leader*

1 Employees could describe an IT problem in Teams and have the service record created, diagnosed and updated. Which support requests would you trust this process to resolve without a technician?

2 The service desk could analyse resolved cases and improve its knowledge each week. Who would own the quality of the knowledge that results?

3 Operations could receive one morning briefing connecting incidents, deployments, cloud changes and business impact. Which decisions would become faster if that view existed today?

4 Monitoring could propose a tested rollback before an engineer opens the cloud console. Which recurring incidents are suitable for that treatment?

5 Every company agent could have its own identity, permissions and expiry date. Who currently owns that control model?

6 Business teams could use approved system capabilities through a conversational layer while IT retains control and the official records remain in ERP, CRM or ITSM. Which capabilities and screens should change first?

7 Every agent action could carry its sources, permissions, approver and result. Could your current audit process reconstruct what an agent did during an incident?

8 Application owners could see unused licenses, overlapping products, support history and renewal deadlines in one continuous view. How much spend reaches renewal without that evidence today?

9 Engineers could delegate migrations, tests and maintenance while concentrating on architecture and customer problems. Is the review process ready for more code arriving faster?

10 The engineering scorecard could measure review load, rework and production stability alongside throughput. Which current metric would give the wrong signal?

11 Junior engineers could learn through structured pairing with senior staff and coding agents. What is your current plan for producing senior engineers in five years?

12 IT could be measured by safe business capacity enabled, while a maintained connector catalog makes each later workflow easier to build. What would that scorecard contain, and which system relationships deserve permanent ownership?

These are the questions the programme is built to answer with the people who run the systems, using their own tickets, code, permissions and operating decisions rather than a model of the function from outside.