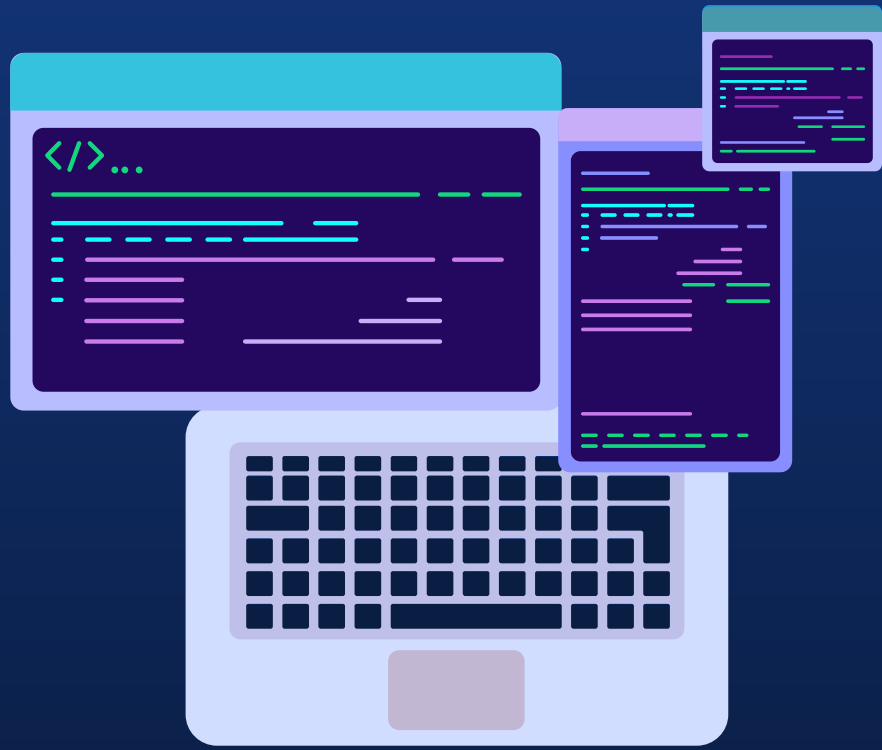
METAPORT

Website maintenance agencies can actually schedule.



Metaport: The Agency Test

Can you search your portfolio as fast as sending an email?

Can you plan for end-of-life dates as easily as planning a sprint?

Are you notified about end-of-life, security, or SSL certificates?

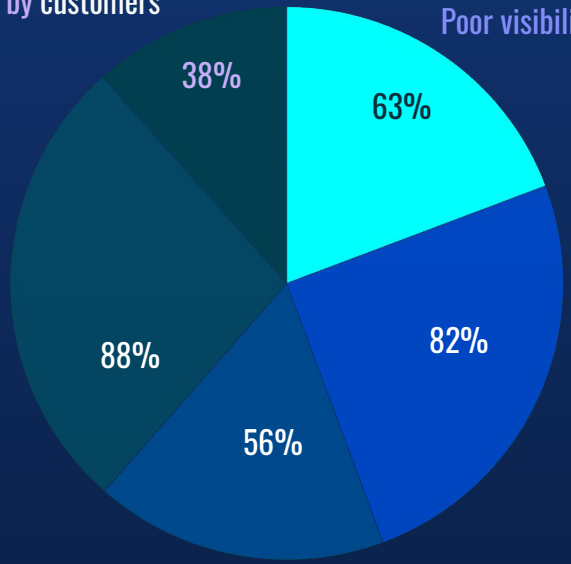
Metaport: Reactive vs Proactive

Minimal end-of-life (EOL) data means agencies cannot lean into **proactive** maintenance. Money is left on the table through **reactive** upgrade projects.

Metaport: Inventory Management

Agency inventory management is manual at best and non-existent at worst. Without a **living** and **macro** portfolio view, agencies plan maintenance **poorly**.

Agency Concessions



source agency survey Feb ‘24 - present

Scenario: Upcoming end-of-life date

Scenario

The team “discovers” a customer’s app whose framework version is end-of-life in 12 months

Risks

- **Client churn** if the problem is not addressed
- **Increased time** (and money) spent supporting apps with unsupported components

Opportunities

- **New revenue from**
  - Planned upgrade pipelines
  - New/renewed maintenance contracts

Next Steps

- **Share** life cycle and roadmap data with customer(s)
- **Update** backlog(s)
- **Locate** ***other applications*** with the same end-of-life date
- **Notify** and plan with the affected customers

Scenario: Security Vulnerability Announcement

Scenario

The team is alerted to three new security vulnerabilities, announced overnight

Risks

- **Client churn** if the problem remains unaddressed
- **Chances of exploit** increase the longer an app is unpatched

Opportunities

- **Increased revenue** by demonstrating what's possible with paid maintenance contracts to customers without one
- **Reduced churn** by demonstrating proactivity to existing customers

Next Steps

- **Share** life cycle and roadmap data with customer(s)
- **Create** ticket(s)
- **Locate** ***other applications*** with the same vulnerabilities
- **Notify** and plan with the affected customers

Scenario: Supply Chain Attack

Scenario

The team “discovers” a supply chain attack in a package manager (and so have your customers!)

Risks

- **Client churn** if the problem remains unaddressed
- **Chances of exploit** increase the longer apps are left unpatched

Opportunities

- **Increased revenue** by demonstrating what's possible with paid maintenance contracts to customers without one
- **Reduced churn** by demonstrating proactivity to existing customers

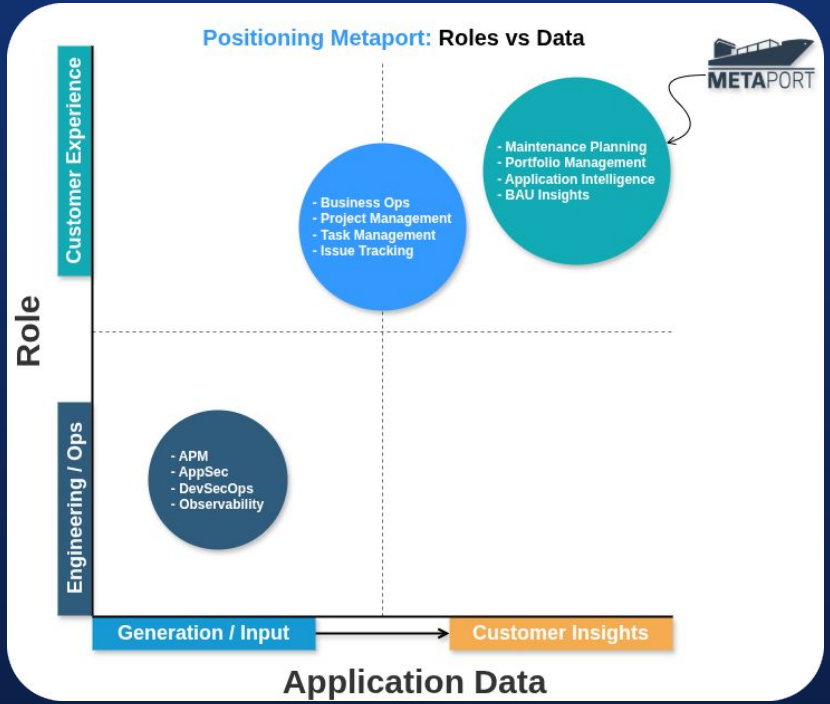
Next Steps

- **Locate** all customer applications which use affected packages and versions
- **Locate** all customer applications which use constraint-based package versioning
- **Create** ticket(s)
- **Notify** and plan remediation with the affected customers (where applicable)

Metaport: Features

- Portfolio-wide end-of-life, security vulnerability, and dependency search
- Shareable application component life-cycles
- Shareable application maintenance calendars
- Custom policy-based, portfolio-wide notifications
- Integrations: JIRA, Dependabot, DependencyTrack
- Multi-tenant with RBAC, MFA, and SSO
- Technology agnostic (PHP, NodeJS, Python, .Net…)
- Metaport CE (Generally Available, Open Source)
- Metaport SaaS (Planned)

Positioning



Summary

Shared data + assets for customer facing roles (engineering & security too)

Self-documenting inventory

Surface applications across portfolios according to end-of-life, security, and dependencies

Curate issues in backlogs as part of your workflows