Project Management Plan

Case Study: Software-Tracking Database for a University

Student: Drew Williams

Introduction

Every university depends on hundreds of software tools, yet few can say exactly where each license lives or when it expires. The IT department needs a reliable way to track what is installed, who uses it, which licenses are active, and when renewals are due. This plan describes how I would lead a small cross-functional team to deliver a secure, searchable database and a lightweight admin UI that the university's IT Asset Management (ITAM) group can maintain long term. The approach follows the six phases in Baars' Project Management Handbook: Initiation, Definition, Design, Development, Implementation, and Follow-up. Breaking work into clear phases reduces risk and makes go or no-go decisions visible at the right moments (Baars, 2006). To strengthen execution, I incorporate modern planning practices that emphasize scope clarity, stakeholder engagement, iterative delivery, and continuous feedback (DeBara, 2023; Joubert, 2021).

1. Initiation

Goal. Assess feasibility and secure sponsorship for a university-wide software-tracking system. From day one we set expectations that we are delivering a working product, not a prototype.

Key questions. Why this project, who is involved, what are the boundaries, and is it feasible. These are the essentials of a sound start (Baars, 2006).

Team and roles.

- Project Lead (me) - owns schedule, budget, and quality
- Sponsor - CIO or Director of IT
- ITAM Manager - business owner and power user
- Security Officer - compliance, FERPA, and data handling
- Systems Engineer - environments and integrations
- DBA - data model and performance
- UX or Front-end Developer - admin UI and usability

Activities. Stakeholder interviews, quick feasibility check on inventory sources (for example SCCM or Jamf exports), define must-have scope, draft initial budget and timeline.

Deliverables. One-page Project Charter with scope, success criteria, boundaries, a rough budget, a high-level timeline, and a go or no-go decision (Baars, 2006).

2. Definition

We translate expectations into clear, testable requirements with the right people at the table, especially future users in ITAM. Projects fail when end users are not engaged and scope is not understood early (Baars, 2006; Joubert, 2021).

Requirements buckets.

- Preconditions: policies, FERPA, vendor license terms

- Functional: CRUD for software, versions, licenses; assign to devices and departments; search and reporting; import feeds
- Operational: target 99.9 percent uptime; search results in under 2 seconds
- Design limitations: SSO required; approved browsers

Activities. Workshops with ITAM, Desktop Support, Procurement, and Information Security. Data discovery of current spreadsheets and device inventories. Define acceptance tests for must-have reports such as “All Adobe installs with license status and renewal date.”

Responsible. Project Lead facilitates, ITAM Manager owns business rules, DBA defines fields and integrity rules, Security Officer validates preconditions, UX Developer shapes admin flows.

Deliverables. Requirements Specification ranked Must, Should, Could; Data Dictionary; Acceptance Criteria; and an updated budget and schedule. Keep the three constraints of scope, resources, and schedule visible and balanced, and define success metrics up front to reduce stress and scope creep (DeBara, 2023).

3. Design

We produce designs that can actually be built and tested. Selecting and documenting a definitive design here reduces downstream churn. Early mockups and dummies help validate choices with users before expensive work begins (Baars, 2006).

Activities.

- Data model: tables for Software, Versions, Licenses, Devices, Departments, Installations with constraints and indexes

- System architecture: PostgreSQL, REST API, SSO, audit logs, backups
- UI mockups: search, detail pages, batch import wizard, license reporting
- Non-functional design: logging, backup and restore targets, basic performance envelope

Responsible. DBA for schema, Systems Engineer for infrastructure, UX or Front-end for flows, Project Lead for trade-offs and decisions.

Deliverables. Design Document with ERD and endpoint list, clickable mockups, and an updated implementation plan. Include a simple roles and responsibilities matrix so everyone knows who decides and who is consulted on key items (DeBara, 2023).

4. Development (pre-build preparation)

Baars treats development here as the staging phase to line up people, suppliers, schedules, and tools so implementation runs smoothly. Small projects may blend this with delivery, but preparation reduces risk (Baars, 2006).

Activities.

- Stand up dev, stage, and prod environments
- Establish CI pipeline and Definition of Done
- Prepare migration scripts and sample datasets
- Confirm integrations and data feeds with owners

Responsible. Systems Engineer manages environments, Project Lead manages schedule and purchasing, DBA finalizes the baseline schema, Security Officer confirms secrets handling and access.

Deliverables. Action Plan and detailed schedule, test data packs, CI or CD checklist, and an access or permissions matrix.

5. Implementation

This is the doing phase: build, test, and demonstrate working software. We execute in short, cyclical iterations of one to two weeks: plan, build, test, and demonstrate. Iteration keeps end users engaged, improves quality, and allows realistic adjustments as knowledge increases (Baars, 2006; DeBara, 2023).

Typical sprints and owners.

- Sprint 1: Core schema, authentication, and audit trail - DBA and back-end
- Sprint 2: Software and Version CRUD with search - back-end and front-end
- Sprint 3: License model and renewals with reports - all
- Sprint 4: Device and installation inventory import - integration focus
- Sprint 5: Department mappings and role-based access - back-end and ITAM
- Sprint 6: Performance tuning and accessibility fixes - all

Quality gates. Unit tests and acceptance tests from earlier phases, accessibility checks, security review, and a data migration rehearsal. Publish tasks and status where everyone can see them to maintain accountability and visibility (DeBara, 2023).

Communication cadence. Weekly standups, biweekly stakeholder demos, and a written progress update that tracks plan versus actual. This supports performance measurement against a baseline for scope, schedule, and cost (Joubert, 2021).

Interim timeline. Estimated six months total: one month for planning and design, three months for iterative build, one month for hardening and testing, and one month for rollout and training.

6. Follow-up

Great projects do not end at ship. Follow-up includes training, documentation, help-desk setup, maintenance plans, post-project evaluation, and a clean team closeout (Baars, 2006).

Activities.

- Knowledge transfer with admin guide, runbooks, ERD, and API docs
- Training for ITAM staff and Service Desk with short videos and reference cards
- Support setup with ticket categories, SLAs, and on-call calendar
- Operations readiness with tested backup or restore and monitoring or alerts
- Closeout with lessons learned, final budget and schedule report, and handoff to operations

Deliverables. User and admin documentation, training materials, operational handoff package, and a final project report. Build a knowledge base so future teams can reuse patterns and avoid repeat mistakes (Joubert, 2021).

Risks and Controls

We continually manage five control factors: time, money, quality, organization, and information (Baars, 2006). Top risks and mitigations:

- Scope creep: use a change control step for any new requirement, and reserve capacity for small changes without endangering must-haves (DeBara, 2023).

- Over-optimistic schedules: create a performance measurement baseline and track plan versus actual each sprint (Joubert, 2021).
- Skipped testing under pressure: enforce Definition of Done and acceptance tests before marking a story complete.
- Resource bottlenecks: identify dependencies early, sequence work, and parallelize only when resources truly allow it (Joubert, 2021).
- Stakeholder misalignment: hold regular demos and feedback loops to keep expectations and reality aligned (DeBara, 2023).

Conclusion

This plan balances structure and flexibility. Baars' six phases provide clear decision points and documentation. Iterative delivery, visible metrics, and active stakeholder engagement reduce risk and keep quality high. The result is a secure, searchable software-tracking database with clear ownership that the university can operate confidently, rather than a design that looks good on paper but stalls in production (Baars, 2006; DeBara, 2023; Joubert, 2021).

References

Baars, W. (2006). *Project management handbook* (Version 1.1). DANS – Data Archiving and Networked Services.

DeBara, D. (2023, July 20). *How to write an effective project plan in 6 simple steps*. Atlassian. https://www.atlassian.com/blog/project-management/write-an-effective-project-plan

Joubert, S. (2021, January 11). *12 steps to develop a project management plan*. Northeastern University Graduate Programs. https://graduate.northeastern.edu/knowledge-hub/developing-project-management-plan/