Delivering a Competitive Microcomputer: Synputer Project Planning and Execution

Module: Software Engineering Project Management

Group 2: Ruben Marques, Lauren Pechey, Victor Angelier, Arianna Poverini, Hristo Todorov

Degree: Masters of Computer Science, University of Essex Online

Module Tutor: Douglas Millward

Date: 01/12/2025

Wordcount: 1098

Table of Contents

Executive Summary & Introduction

The Synputer project - originating from Colin Syn and Will Burns and refined through subsequent technical analysis - aims to deliver a competitive microcomputer within a £500,000 budget and 13-month schedule. Achieving the objectives with a focus on industry-standard compatibility and business functionality by defining feasible requirements, resolving stakeholder conflicts, and establishing a realistic architecture that aligns technical ambition with commercial and operational constraints.

Methodology

A Hybrid Agile Software Development Life Cycle integrates Scrum sprints with formal stage-gates to manage hardware-software complexity while maintaining flexibility and governance. Behaviour-driven development ensures clear, testable requirements, reducing ambiguity and rework while maintaining control over cost, schedule, and hardware–software integration risks.

Key Requirements

Revised requirements prioritise feasibility: industry-standard OS, removable storage, expansion capability, TeleBasic compatibility, and essential connectivity. Core technical foundations include a Motorola 68k CPU, 512KB RAM, and desktop format, while high-risk features such as portability and proprietary technologies are removed to meet budget and time limits.

Plan and Budget

The 13-month plan allocates time for design, assembly, integration, testing, and refactoring, supported by iterative sprints and stage-gated oversight. Budget distribution—hardware, software, labour, and contingencies—ensures disciplined resource use and supports reliable delivery within financial and temporal constraints.

Stakeholder Requirements, Assumptions & Gaps

The initial requirements extracted from the transcript between Colin Syn (Synful) and Will Burns (EDC) establish expectations on portability, backward compatibility, and affordability (Kancharla, 2025). **Table 1** summarises extracted requirements alongside corresponding success criteria, providing a structured baseline for evaluating system feasibility and performance targets.

Task	Extracted Requirements (Findings/Analysis)	Success Criteria (HW & SW)
Extract Requirements	**Hardware:** Motorola 68k CPU; ≥512KB RAM; expandable memory; solid-state storage; 4-inch monochrome display; serial/keyboard/external display/joystick ports; <2kg weight; ≥2-hour battery life. **Software:** Unix-like multitasking OS; bundled productivity suite; HyperBasic with TeleBasic converter; affordability and backward compatibility.	**Hardware**: Device ≤2kg; ≥2-hour battery; compatible with business peripherals. **Software:** Stable Unix-like OS; bundled apps run concurrently; HB converter runs TeleBasic programs with no functional errors; project meets £500,000 budget and 13-month timeline.

Facilitate HW/SW Integration	Align hardware interfaces with OS and HyperBasic; define emulator performance; verify I/O interactions.	System boots reliably; OS loads; HB and productivity apps run concurrently (Koelsch, 2023).
Translate Requirements into User Stories	User stories emphasise portability, performance, compatibility, and usability.	Targets met: ≤2kg device, ≥2-hour battery, HB backward compatibility demonstrated.
Resolve Sprint Retros & Blockers	Clarify TeleBasic licensing; obtain missing OS documentation; unblock development tasks.	All blockers resolved before sprint closure.
Validate UAT Readiness	UAT includes HB converter testing, hardware validation, stress testing, and usability checks.	All user stories pass acceptance criteria; prototype meets benchmarks; EDC approval granted (Koelsch, 2023).
Contribute to Project Report	Produce requirements matrix, risk assessment, and traceability evidence from requirements to UAT.	Clear, auditable documentation that meets governance standards.

Table 1: Extracted Requirements & Success Criteria

In contrast, many elements of the requirement set rely on unverified assumptions—such as the availability of a Unix-like OS, the feasibility of TeleBasic-to-HyperBasic conversion, and reliance on existing Synful architecture. These areas introduce uncertainty and potential risk within the strict £500,000 budget and 13-month delivery window (Koelsch, 2023). Addressing uncertainties early is essential to ensure the proposed Motorola 68k-based portable system can be delivered reliably and integrated successfully (Marnada et al., 2022). **Table 2** details the key assumptions and gaps identified.

Task	Assumptions & Gaps Identified
Extract Requirements	No confirmed Unix OS supplier; unclear licensing; no battery or QA specifications; informal vendor relationships; missing performance metrics; undefined target market (Koelsch, 2023).
Document Assumptions & Gaps	Reliance on Synful architecture; informal agreements with software vendors; no integration protocol documentation; no testing strategy.
Facilitate HW/SW Integration	Dependencies between OS, HB interpreter, and hardware drivers require validation (Koelsch, 2023).
Translate Requirements into User Stories	Weight, CPU power, and storage performance not yet validated.
Resolve Sprint Retros & Blockers	Vendor delays may affect OS or hardware milestones (Kancharla, 2025).
Validate UAT Readiness	Prototype hardware and software must exist before UAT; schedule risk if components slip.
Contribute to Project Report	Documentation updates reliant on timely clarification from suppliers and engineering teams.

Table 2: Assumptions & Gaps

Proposed Development Methodology

Methodology Statement – Synputer Development Project

The Synputer project demands an integrated hardware–software lifecycle within strict constraints (thirteen months, £500.000 budget). Pure Waterfall models lack agility for rapidly evolving firmware (Ford et al., 2025); conversely, pure Agile cannot accommodate fixed hardware lead-times typical of component procurement (Andres, 2025; RayMing PCB, 2024). The team therefore adopted a Hybrid Agile SDLC combining two-week Scrum sprints with formal stage-gates for design and integration. This provides flexibility for software innovation whilst maintaining the rigour required for hardware manufacturing.

Structure:

- *Requirements & Feasibility Gate* – Stakeholder needs validated against Hardware and Software BOMs.
- *Design & Architecture Gate* – System architectures peer-reviewed before procurement.
- *Incremental Development Sprints* – Two-week sprints deliver firmware/UI modules, ending with sprint review (60 min) and retrospective (90 min).
- *Integration & Test Gate* – Hardware prototypes and software builds verified via Behaviour-Driven Development tests.
- *User Acceptance & Handover Gate* – Final validation and release.

Justification:

This model delivers both flexibility (Agile iterations) and governance (stage-gates). Bi-weekly retrospectives are essential: fixed-budget projects tolerate no waste. Durham and Michel (2021) emphasise that Lean feedback loops eliminate inefficiency; Agrawal et al. (2024) show 31 per cent of design errors stem from knowledge gaps, which structured retros mitigate through same-sprint corrective action. Brooks (1975) warns that late project expansion delays delivery—our stable, cross-functional teams avoid this trap. ISO/IEC/IEEE 16326 (2019) mandates defined gates for auditability; Scrum reviews allow stakeholder inspection, preventing uncontrolled scope creep. Retrospectives are vital for hardware–software integration: when pin-mapping or ULA–CPU design assumptions fail, a retro allows same-sprint mitigation instead of costly redesigns months later.

Iterative sprints shorten feedback cycles whilst gate reviews and retrospectives maintain accountability, knowledge capture, and cost control, directly supporting delivery within fixed financial and temporal constraints. For exam[ple, in Sprint 2, a retro revealed the HyperBasic compiler failed on 512 KB systems (knowledge-gap error). A Definition-of-Done update enforced memory-constrained testing. This four-hour fix prevented a week-long redesign, demonstrating clear ROI.

Revised Requirements

Analysis of gathered requirements against technical feasibility, stakeholder priority and constraints reveals significant gaps between initial vision and realistic delivery (Ali et al., 2020).

Weight and battery targets are unachievable: the 2kg portable specification cannot be met (minimum viable hardware requires 3.35kg), and battery life delivers only 15-20 minutes against a 2 hour requirement.

Stakeholder conflict exists between Syn's portability preference and Burns' requirement for industry standard compatibility. EDC's takes precedence given £500,000 funding and the business market focus. Risk concentration is evident in assigning Synful's small team (1 architect, 2 engineers) responsibility for multiple complex features within 13 months.

A critical GUI gap: bundled graphics applications require GUI, yet available OS are CLI-based, making graphics packages non-functional and office tools severely limited. Resolving this is essential for commercial viability.

Table 3 presents revised requirements using MoSCoW prioritisation (Kravchenko et al., 2022), establishing criteria for project success.

FUNCTIONAL REQUIREMENTS			
Requirement	**Priority**	**Details**	**Justification**
Industry Standard OS	**Must Have**	The system must run a third-party, industry-standard, multi-tasking OS (e.g., MccOS or 68kDOS).	Stakeholder Critical: Main requirement from the primary financial backer (EDC, which resolves the core strategic conflict.
Standard Removable Drive	**Must Have**	The system must include at least one industry-standard removable drive (e.g., a 3.5" floppy drive).	Stakeholder Critical: A non-negotiable requirement from EDC to ensure business data can be exchanged with other systems.
Expandable System	**Must Have**	The system architecture must allow for the addition of expansion packs via slots.	Important for future-proofing and a key feature of competitors (exact number/type of need negotiation to meet cost targets.
External Keyboard Support	**Must Have**	The unit must have a dedicated port for an external keyboard, and a physical keyboard will be supplied.	Stakeholder Critical: A firm requirement from Will Burns for a business machine; its absence would be a breach of contract.
TeleBasic Compatibility	**Must Have**	The system must run existing TeleBasic programs without	Stakeholder Critical: A key requirement from EDC to ensure a seamless migration path for their existing customers.

		requiring users to manually convert or rewrite them.	
Bundled Office Suite	**Must Have**	A complete office productivity suite will be licensed and pre-installed on the system.	Value Proposition Critical: The key software feature that makes the Synputer a complete 'out-of-the-box' business solution.
Multi-Connectivity	**Must Have**	The system must provide a minimum of 2 serial ports and an external display connector.	Stakeholder Critical: These are specific technical requirements from EDC for business networking and peripheral support.
NON FUNCTIONAL REQUIREMENTS			
Requirement	**Priority**	**Details**	**Justification**
Desktop Form	**Must Have**	The system will be designed in desktop format	Technically Necessary: Portable concept with <=2kg of weight is impossible to achieve.
Minimum 512KB RAM	**Must Have**	The motherboard will be equipped with at least 512KB of RAM.	Performance Critical: A non-negotiable technical specification from EDC, required to run the target multi-tasking OS effectively.
Motorola 68k Series CPU	**Must Have**	The system will be powered by a CPU from the Motorola 68k family.	Technically Foundational: A core technology decision agreed by both stakeholders that dictates the machine's performance and future compatibility.
GUI Interface	**Should Have**	A plan must be developed to port or create a GUI	Usability: Necessary to make the bundled office suite fully functional and meet modern user expectations.

		environment for the chosen OS.	
Game Emulator	**Could Have**	An application that allows the Synputer to play games from older Synful Computing machines.	A secondary goal that serves the retail market. This can be deferred to a post-launch software release to reduce initial project risk and cost.
Joystick Port	**Could Have**	A port to connect a standard joystick for gaming.	A desirable feature for the secondary gaming market, but not essential for the core business product. Can be added via an expansion card.
Portable Form Factor	**Won't Have**	The initial concept for a portable machine with a built-in screen and battery.	Proven to be technically impossible by the weight and battery life data in the case study.
In-House OS / HyperBasic	**Won't Have**	The plan to develop a proprietary OS, programming language, and converter from scratch.	Represents an unacceptable level of risk to the project's timeline and budget
Proprietary Solid-State Storage	**Won't Have**	The plan to invent a new solid-state storage medium.	An unproven, high-risk R&D effort that is not feasible within the project's constraints.

Table 3: Functional & Non-Functional Requirements

Gherkin's Behavioural Scenarios

To translate high-priority functional requirements into testable system behaviour, Gherkin-based acceptance scenarios were created (Budihal, 2025). These scenarios specify system interactions using the BDD 'Given–When–Then' structure, ensuring clarity, traceability, and alignment between stakeholders, developers and testers (Budihal, 2025), as shown in **Table 4.**

Task / Requirement Area	**User Stories (Business / Functional Perspective)**	**Gherkin Scenarios (Testable Specifications)**
		FEATURE: Synputer Hardware Requirements ***The Synputer must meet hardware specifications to ensure portability, performance, and compatibility for business and developer use.***
CPU Compatibility	As a developer, I want the system to run applications on the Motorola 68k CPU so software executes reliably.	Scenario 1: Verify CPU compatibility Given the system uses a Motorola 68k CPU When an application is executed Then it should run correctly on the hardware And be forward-compatible with later 68k series chips
RAM / Performance	As a user, I want 512KB of RAM so I can run multiple applications without slowdowns.	Scenario 2: Validate minimum RAM Given the system has 512KB of RAM When running multiple applications Then the system should not crash or slow below acceptable limits
Storage	As a user, I want fast and expandable storage so I can store and retrieve files efficiently.	Scenario 3: Check storage medium Given solid-state storage is installed When saving or retrieving files Then the system must complete the operation within acceptable time limits And allow expansion with standard drives if required

Screen / Display	As a user, I want a clear 4" display so I can read text and view graphics easily.	Scenario 4: Test screen and display Given a 4-inch low-power monochrome screen When displaying text or graphics Then the content must be readable and refresh correctly
Battery Life	As a mobile user, I want the system to last at least 2 hours so I can work without frequent charging.	Scenario 5: Validate battery life Given the system is fully charged When operating under normal load Then it should run for a minimum of 2 hours
Connectivity	As an office manager, I want multiple ports so all peripherals work seamlessly.	Scenario 6: Test connectivity ports Given serial, keyboard, external display, and joystick ports When peripherals are connected Then all ports should function as expected
		FEATURE: Synputer Software Requirements and UAT *The system must meet software specifications, backward compatibility, and pass end-to-end testing to ensure usability and business functionality.*
Bundled Software	As a user, I want a complete business suite so I can perform all office tasks.	Scenario 7: Verify bundled business suite Given the system includes a word processor, spreadsheet, database, and graphics software When a user opens the suite Then all applications should launch successfully and allow standard operations
HyperBasic Compatibility	As a developer, I want legacy TeleBasic programs to run on HB so migration is smooth.	Scenario 8: Test HyperBasic (HB) backward compatibility Given a TeleBasic program When the program is converted using HB converter Then it should execute correctly on the Synputer without manual intervention

Legacy Game Emulation	As a user, I want old games to run so I can use the system for both work and leisure.	Scenario 9: Validate emulation of legacy games Given a game designed for older Syn machines When run on the Synputer Then it should operate at full speed alongside a business application
Portability	As a user, I want a lightweight device so I can carry it easily.	Scenario 10: Ensure device portability Given the system includes all components (CPU, battery, screen, drives) When lifted or transported Then the total weight should not exceed 2kg
Third-Party Unix OS	As an engineer, I want integration with Unix OS so the system supports multiple software environments.	Scenario 11: Integration of third-party Unix OS Given the in-house system interfaces with a third-party Unix OS When the OS is installed Then the system should boot and run both OS and bundled applications correctly
UAT Readiness	As a QA tester, I want all features validated so the system is ready for end-users.	Scenario 12: Validate system UAT readiness Given the completed hardware and software When performing end-to-end testing Then all acceptance criteria for performance, compatibility, and usability are met

Table 4: Gherkin Statements

Project Plan and Budget

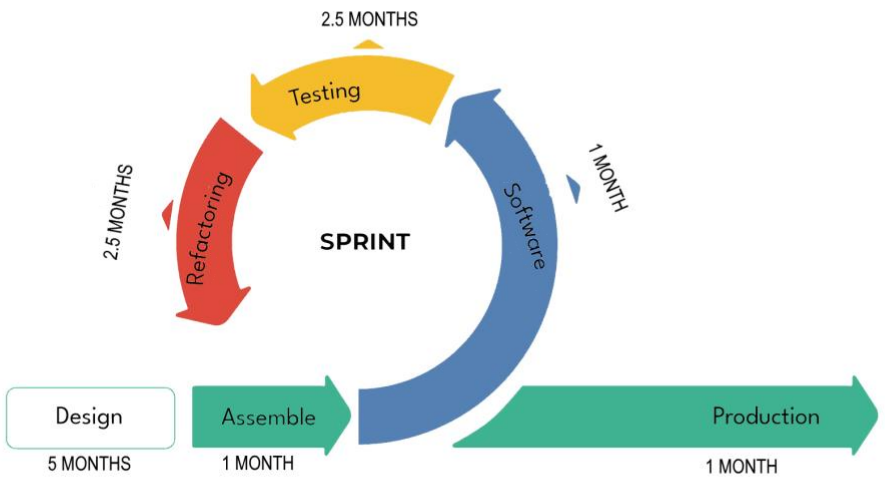


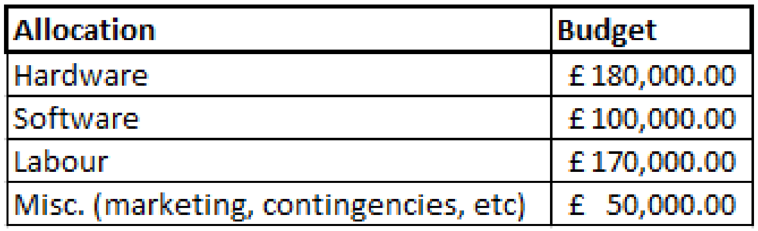
Figure 1: Project Timeline

The timeline above details milestones and their budgeted timeframes. The design includes hardware architecture, as well as the licensing of software such as the Office Suite. Hardware assembly and software development/integration are budgeted to take less time, with a larger focus on testing and refactoring (Koelsch, 2023).

The format of the timeline is a Sprint Lifecycle chart, expecting that some iteration after testing may be necessary. The total time spent per milestone is budgeted for the 13 month total deadline, and includes leeway time in case it is needed. Testing and refactoring time take around 20% of the time budget each to ensure proper integration. Testing is also performed in parallel to the first 7 months of the plan as detailed below in **Table 5** and **6**.

Type	**Justification**	**Type of testing**	**Duration**	**Related**
Hardware testing	Meet the performance requirements	Performance, Integration	1 month	Scenario 1 Scenario 3 Scenario 4
Software test (OS, Office Suite)	Find bugs in early stages	Integration, Performance, UX, End2End	1 month	Scenario 2 Scenario 6 Scenario 7 Scenario 9
Contingency	When blocking issues are found.	Fix UX issues if present	0.5 month	Scenario 5 Scenario 9 Scenario 10 Scenario 12
OS testing (**Software**) TDD/BDD	Assure requirements are met before integrating into hardware	Feature + Unit testing + Profiling (Automated)	4.5 months	Scenario 2 Scenario 4 Scenario 8 Scenario 11

Table 5: Parallel testing methods (during design, assembly and software integration)



Allocation	**Budget**
Hardware	£ 180,000.00
Software	£ 100,000.00
Labour	£ 170,000.00
Misc. (marketing, contingencies, etc)	£ 50,000.00

Table 6: Budget and Price Allocation

Through commissioned unit cost of £250 and an advertised price of £399.99, the plan ensures positive revenue of £299,980 and a clear path toward a break-even point (Sintha, 2020).

Conclusion

The original Synputer vision required significant refinement to meet budget, schedule, and technical constraints. By reassessing the requirements, resolving conflicts, and adopting a Hybrid Agile SDLC, the project team established a realistic, testable, and stakeholder-aligned plan. The revised requirements ensure feasibility, while Gherkin specifications provide clear validation criteria. Together, the clarified scope, structured methodology, and controlled risk management create a viable path toward delivering a competitive, industry-standard microcomputer within the £500,000 budget and 13-month timeframe.

References

Agrawal, T., Walia, G.S. and Anu, V.K. (2024) Development of a software design error taxonomy: a systematic literature review. *SN Computer Science*, 5(467). doi: 10.1007/s42979-024-02797-2.

Ali, A., Hafeez, Y., Hussain, S., & Yang, S. (2020) Role of requirement prioritization technique to improve the quality of highly-configurable systems. *IEEE Access*, 8, 27549-27573. doi: https://doi.org/10.1109/ACCESS.2020.2971382

Andres, J. (2025) 'Agile Approach Applied to Hardware Development', LinkedIn Pulse, 22 March. Available at: https://www.linkedin.com/pulse/agile-approach-applied-hardware-development-jerome-andres-gy2de (Accessed: 25 November 2025).

Brooks, F.P. (1975) *The Mythical Man-Month: Essays on Software Engineering.* Reading, MA: Addison-Wesley. Available via Open Library.

Budihal, A. (2025) *A Guide for Implementing the Behavior-Driven Development (BDD) Framework in IT Projects*. Bachelor's/Master's thesis. Available at: https://www.theseus.fi/handle/10024/890621

Durham, D. and Michel, C. (2021) *Lean Software Systems Engineering for Developers: Managing Requirements, Complexity, Teams, and Change Like a Champ.* 1st edn. Berkeley, CA: Apress. ISBN 978-1-4842-6932-9. Available at: SpringerLink.

ISO/IEC/IEEE (2019) *16326: Systems and Software Engineering — Life-Cycle Processes — Project Management.* Geneva: International Organization for Standardization. Available at: ISO Catalogue.

Ford, [Initial], [et al.] (2025) 'Bridging the Gap: Aligning Waterfall and Agile Approaches', PM World Journal, 155, August 2025. Available at: https://pmworldlibrary.net/wp-content/uploads/2025/08/pmwj155-Aug2025-Ford-et-al-Bridging-the-Gap-Aligning-Waterfall-and-Agile-2.pdf

Gemino, A., Horner Reich, B., & Serrador, P. M. (2020) Agile, Traditional, and Hybrid Approaches to Project Success: Is Hybrid a Poor Second Choice? *Project Management Journal*, *52*(2), 161-175. doi: https://doi.org/10.1177/8756972820973082

Kancharla, V. (2025) Enhancing Software Development: Strategies For Achieving Code Portability Across Technology Platforms. *International Journal of Information Technology & Management Information System* 16(1): 6-21. DOI: https://doi.org/10.34218/IJITMIS_16_01_002

Koelsch, G. (2023) *Hardware and Software Projects Troubleshooting: How Effective Requirements Can Save the Day.* 2nd ed. Berkeley, CA: Apress. DOI: https://doi.org/10.1007/978-1-4842-9830-5.

Marnada, T., Raharjo, T., Hardian, B., & Prasetyo, A. (2022) Agile Project Management Challenge in Handling Scope and Change: A Systematic Literature Review. *Procedia Computer Science*, 197(1), pp. 290–300. DOI: https://doi.org/10.1016/j.procs.2021.12.143.

Papadakis E., Tsironis L.K. (2020) ‘Towards a Hybrid Project Management Framework: A Systematic Literature Review on Traditional, Agile and Hybrid Techniques’. Available at https://journalmodernpm.com/manuscript/index.php/jmpm/article/download/JMPM02410/394

RayMing PCB (2024) 'Agile for Hardware: Sprints and Working Prototypes', Printed Circuit Board, 15 April. Available at: https://www.printedcircuitboard.org/2024/04/agile-for-hardware-sprints-and-working.html (Accessed: 25 November 2025).

Sintha, L. (2020) ‘Importance of Break-Even Analysis for the Micro, Small and Medium Enterprises‘, *International Journal of Research - Granthaalayah*, 8(6). Available at: http://repository.uki.ac.id/2044/14/ImportanceofBreakEven.pdf (Accessed 27 November 2025).

Vijayakumar, S., Prasad, K.K. and Holla, R.M. (2024) 'Assessing the effectiveness of MoSCoW prioritization in software development: a holistic analysis across methodologies', *EAI Endorsed Transactions on Internet of Things*, 10(1). doi: https:doi.org/10.4108/eetiot.6515

Zasa, F. P., Patrucco, A., & Pellizzoni, E. (2020). Managing the Hybrid Organization: How Can Agile and Traditional Project Management Coexist? *Research-Technology Management*, *64*(1), 54–63. Available at https://doi.org/10.1080/08956308.2021.1843331