Illustrating the Use of the Model

Part 1: Model Narrative

Type of Project

The software being estimated is a **Smart Parking Meter System**—an embedded application designed to automate public street parking by enabling contactless payment, real-time availability updates, and communication with a centralized management system. Each meter is equipped with an array of sensors to detect vehicle presence, a display screen and payment module (card, NFC, or app-based), and wireless communication to interact with backend services. This software must run reliably in outdoor environments under extreme weather, with limited access to power or maintenance. Given its reliance on hardware integration, tight constraints, and mission-critical reliability, the system fits squarely within the **Embedded** mode of the COCOMO model as defined by Boehm [1].

Mode and Size

Mode: This project fits the Embedded Mode. The system must interact with dedicated hardware, meet strict real-time and environmental constraints, and work reliably with minimal intervention.

Size (SLOC): I estimate the software will contain around 30,000 Source Lines of Code (SLOC), or 30 KLOC. The size estimate includes code for sensor drivers, user interface handling, secure payment processing, network protocols, telemetry reporting, diagnostic logging, and over-the-air update mechanisms.

Project Factors

The following cost drivers were chosen based on realistic assumptions about the team, environment, and the product:

Category	Driver	Value	Reason
Product	REL	H	High reliability is needed in outdoor unattended environments
	DATA	N	Standard input/output, small DBs for sync.

	CPLX	H	Moderate complexity for real-time validation + reporting.
Computer	TIME	H	Real-time processing required for sensor & payment handling.
	STOR	N	Nominal storage requirement (local + cloud).
	VIRT	L	Minimal use of virtual machines.
	TURN	N	Moderate turnaround due to iteration cycles.
Personnel	ACAP	H	Highly skilled team with embedded experience.
	AEXP	H	Developers experienced in embedded systems.
	PCAP	H	Above-average programmer capability.
	VEXP	N	Moderate experience with tools.
	LEXP	N	Familiarity with the language used (C/C++)
Project	MODP	H	Good software tools and modern practices used.
	TOOL	H	Using automated CI/CD, debuggers, profilers.
	SCED	N	The schedule is realistic, not rushed.

Estimation Report

After inputting these values into NASA's online COCOMO calculator [3], the results are as follows:

- Effort: 137.260 Person-Months
- Duration: 12.077 Months
- Recommended Team Size: 11.366

These results were calculated using Boehm's original model coefficients adapted for embedded systems:

- a = 2.317, b=1.2, c=25, d= 0.32

These figures reflect the expected tradeoff between performance, reliability, and real-time constraints versus the productivity boosts from experienced personnel and strong tool support. Boehm's equation, Effort = a * KLOC^b, was applied using the embedded mode coefficients a =

2.317, b = 1.2, while schedule estimation used Duration = c * Effort^d, with c = 2.5, d = 0.32 [1][2].

Part 2: Worst Case Scenario

Cost Drivers Adjustments

In this scenario, the project team faces severe operational and organizational challenges. All cost drivers were adjusted to reflect **worst-case values**. This includes the selection of very low analyst and programmer capabilities, virtually no application experience, extremely limited tool usage, unstable development platforms, and an unreasonably compressed schedule. Such conditions are common in rushed procurement-based projects with little upfront planning or training.

- Personnel Capability and Experience: Set to Very Low across analyst, programmer, and application experience, reflecting a newly assembled team unfamiliar with embedded systems.
- Tool Support: Set to Very Low, implying no automation, outdated IDEs, and manual testing.
- Schedule Constraint: Very Low — indicating a rushed timeline with no testing buffer.
- Platform Volatility: Very High — hardware changes during development add further instability.
- Reliability Requirement: Very High — legal and safety compliance demands strict reliability, adding to the burden.

Estimation Report

Category	Driver	Value
Product	REL	VH
	DATA	VL
	CPLX	VH
Computer	TIME	VH
	STOR	VH

	VIRT	VH
	TURN	VL
Personnel	ACAP	VL
	AEXP	VL
	PCAP	VL
	VEXP	VL
	LEXP	VL
Project	MODP	VL
	TOOL	VL
	SCED	VL

COCOMO output (hypothetical worst-case approximation):

- Effort: 4528.018 Person-Months
- Duration: 36.968 Months
- Recommended Team Size: 122.483 Developers
- Adjusted ‘a’ Coefficient: 76.447

These values were generated using the formula [3]:
Effort = a × KLOC^b, where a = 76.447 and b = 1.2.
Duration = c × Effort^d, where c = 2.5 and d = 0.32.

Compared to the nominal scenario (137.260 PM and 12.077 months), effort **increased by more than 30×** and the timeline **tripled**. This reflects how detrimental project risks like inexperience, high system volatility, and poor tooling can be. Boehm's model [1][2] validates that such poor conditions drastically increase the adjusted effort factor, especially under embedded system constraints. The unrealistic staffing demand (122+ developers) further highlights that throwing more people at a delayed project often worsens the situation. In real-world terms, such a project would almost certainly fail or result in massive budget overruns [4].

Part 3: Ideal Conditions Scenario

Cost Drivers Adjustments

In contrast, this scenario represents a highly optimized development environment. All relevant cost drivers were set to reflect **Very High or optimal values**. The team consists of expert analysts and developers with deep embedded system experience. Full automation, advanced static and dynamic analysis tools, robust version control, test harnesses, and performance profiling are all in use. Hardware is stable, the development cycle is well-planned, and schedule constraints are minimal.

For example, Required Reliability and Product Complexity were both adjusted downward, reflecting clean modular design and reuse of tested components. Analyst and Programmer Capability were set to Very High, meaning team members are expected to write clean, efficient, and bug-resistant code from the outset. Use of Modern Programming Practices and Tools was set to Very High, indicating best-in-class development workflows [4].

- Personnel Attributes: Very High — expert developers and analysts with deep platform and application experience.
- Tool Support and Modern Practices: Very High — complete CI/CD, static and dynamic testing, and high automation.
- Schedule Constraint: Very High — ample time to deliver with iteration cycles.
- Product Complexity and Reliability: Low/Moderate — due to improved modular design and reuse of components.

Estimation Report

COCOMO output (hypothetical best-case approximation):

Category	Driver	Value
Product	REL	L
	DATA	VL
	CPLX	L
Computer	TIME	L
	STOR	L
	VIRT	VL
	TURN	VH

Personnel	ACAP	VH
	AEXP	VH
	PCAP	VH
	VEXP	VH
	LEXP	VH
Project	MODP	VH
	TOOL	VH
	SCED	VH

- Effort: 39.131 Person-Months
- Duration: 8.082 Months
- Recommended Team Size: 4.841 Developers
- Adjusted ‘a’ Coefficient: 0.6606

These results were calculated using [3]:
Effort = a × KLOC^b, where a = 0.6606 and b = 1.2.
Duration = c × Effort^d, where c = 2.5 and d = 0.32.

Compared to the base case (137.260 PM), the ideal scenario reduces effort by over 70%, and shortens the schedule by 4 months. This outcome illustrates the substantial impact of high team capability, strong toolchains, and well-structured project management. Studies confirm that programmer capability is the most influential cost driver in COCOMO [2], and this report supports that observation. While such perfect conditions are rare, organizations can achieve significant gains by investing in team training, stable architecture, and development infrastructure [4].

References

[1] B. Boehm, COCOMO II Model Definition Manual (1995) *Center for Software Engineering, USC*. Available at https://www.rose-hulman.edu/class/cs/csse372/201310/Homework/CII_modelman2000.pdf (Accessed: 20 April 2025)

[2] B. W. Boehm, Software Engineering Economics (1983) *ACM Sigsoft Software Engineering Notes*. Available at https://dl.acm.org/doi/pdf/10.1145/1010891.1010897 (Accessed: 20 April 2025).

[3] NASA STRS COCOMO Calculator (2025) *Glenn Research Center.* Available at https://strs.grc.nasa.gov/repository/forms/cocomo-calculation/ (Accessed: 20 April 2025).

[4] R. S. Pressman, Software Engineering A Practitioner's Approach 7th Edition (1981) *McGraw Hill*. Available at https://mlsu.ac.in/econtents/16_EBOOK-7th_ed_software_engineering_a_practitioners_approach_by_roger_s._pressman_.pdf (Accessed: 20 April 2025).