**Illustrating the Use of the Model**

# **Part 1: Model Narrative**

***Type of Project***

Based on "Rust for the IoT" by Joseph Faisal Nusairat, this project implements an embedded home automation system using Rust on Raspberry Pi.

**Key Features:**

* **Sense HAT Integration**: Utilizes Sense HAT for sensor data collection (temperature, humidity, pressure, gyroscopic data).
* **Raspberry Pi Camera Module**: Captures high-definition video footage for motion detection.
* **Localized Video Storage**: Stores video locally for privacy and data control, with no remote
* **iOS Integration**: Allows iOS device integration for motion sensor notifications and temperature monitoring via the Home App.
* **Modularity**: Designed for easy expansion with additional sensors or devices.
* **Automated Testing**: Implements automated testing for reliability assurance.
* **Docker Build**: This uses Docker to run the tests in the target environment.

***Mode and Size***

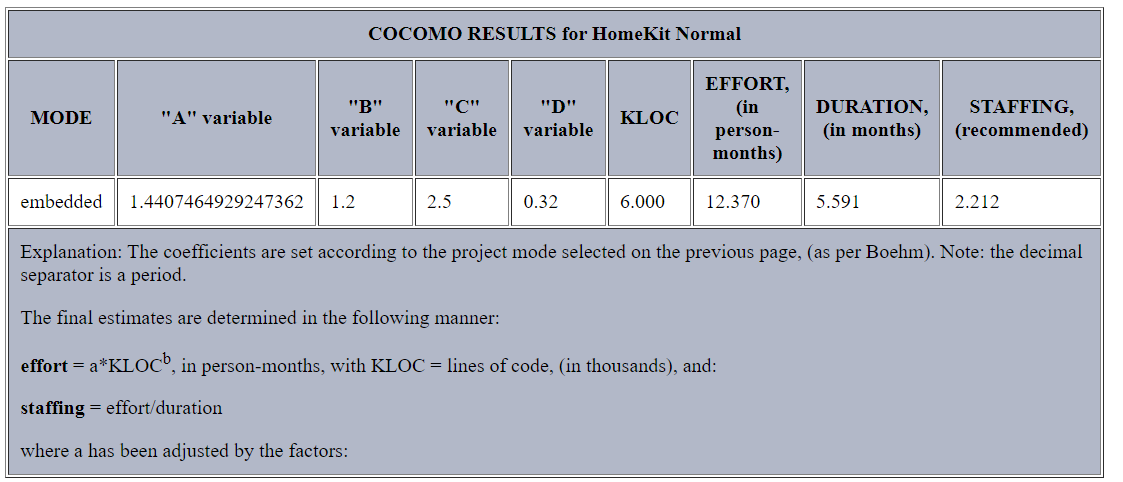
The selected mode for this project is Embedded mode due to its necessity to operate within stringent hardware, software, and operational constraints.

* Base Functionality: The system's core functionality will comprise approximately 2000 lines of code (LOC). The estimate encompasses tasks such as interfacing with hardware components (Sense HAT, Raspberry Pi Camera Module), processing sensor data, managing video capture and storage, integrating with the Home App, and handling system logic.
* Test Code: An additional 4000 lines of code (LOC) are needed for testing purposes. This estimate includes unit tests for individual components, integration tests for system interactions, and end-to-end tests for critical user scenarios.

***Project Factors***

|  |  |  |
| --- | --- | --- |
| **Project Factor** | **Value** | **Explanation** |
| **Product Attributes** | | |
| Required Reliability | N | In failure, the user can restart and recover without losing any information. |
| Database Size | N | The database size estimate is 300 KB, which sets the D/P to 50. |
| Product Complexity | L | Integrating with sensors will not require complex predicates, and there is no need for heavy code nesting. |
| **Computer Attributes** | | |
| Execution Time Constraint | N | Software must operate continuously and likely use less than 50% of the available CPU. |
| Main Storage Constraint | N | We will likely use 50% or less of available space for video storage, which we must delete once continuously when it becomes too old. |
| Platform Volatility | L | Raspberry Pi 4 is a reliable platform with minimal changes. |
| Computer Turnaround Time | L | Users must receive updates when they happen on the Application; it must be interactive. |
| **Personnel Attributes** | | |
| Analyst Capability | H | Writing relevant automated test scenarios and code-matching the required features is essential. |
| Applications Experience | L | It's a straightforward application that does not require much UX design. |
| Programmer Capability | N | The complex part of the Application is integrating devices, which means we will need a capable programmer. |
| Platform Experience | H | Integrating sensors with the programming language, the platform, and how to deploy and test requires some experience. |
| Programming Language and Tool Experience | H | Rust is not a beginner-friendly language, and deploying and automated testing are equally complex. We will need someone with previous experience with the tools. |
| **Project Attributes** | | |
| Modern Programming Practices | VH | Rust enforces strict best practices, which means we get this factor with the language. |
| Use of Software Tools | VH | The project requires a Unit Testing framework, Package Manager, Deployment Environment, and Integration with iOS and Raspberry Pi 4. Library improvements might be necessary for integrating sensors. |
| Required Development Schedule | N | Normal pace development. |
| **New** | | |
| Required reusability | L | No reusable components are required. |
| Documentation match to life-cycle needs | H | Software design documentation is needed to back up the project when new features are required. |
| Personnel continuity | N | Same engineer continously. |
| Multisite development | N | 100% Local. |

***Estimation Report***



# **Part 2: Worst Case Scenario**

***Cost Drivers Adjustments***

|  |  |  |
| --- | --- | --- |
| **Project Factor** | **Value** | **Explanation** |
| **Product Attributes** |  |  |
| Required Reliability | VH | User had major financial loss due to lack of motion detection. |
| Database Size | VH | Database increased in size without cleaning jobs. |
| Product Complexity | VH | Queue handling to communicate with HomeKit increased complexity |
| **Computer Attributes** |  |  |
| Execution Time Constraint | H | Video Processing was more intensive than anticipated. |
| Main Storage Constraint | H | Video quality took more space, or space capability was reduced. |
| Platform Volatility | H | Raspberry Pi 4 had major security issues. |
| Computer Turnaround Time | L | Still needs to be interactive. |
| **Personnel Attributes** |  |  |
| Analyst Capability | VL | Our personnel is new to everything, and the project is not in good shape. |
| Applications Experience | VL |  |
| Programmer Capability | VL |  |
| Platform Experience | VL |  |
| Programming Language and Tool Experience | VL |  |
| **Project Attributes** |  |  |
| Modern Programming Practices | VL |  |
| Use of Software Tools | VL |  |
| Required Development Schedule | VL |  |
| **New** |  |  |
| Required reusability | L | No reusable components are required. |
| Documentation match to life-cycle needs | VL | Software design documentation is needed to back up the project when new features are required. |
| Personnel continuity | VL | Same engineer continously. |
| Multisite development | VH | 100% Local. |

***Estimation Report***

A screenshot of a test results

Description automatically generated The estimate of 2 staff for the project escalates to 34 due to critical factors across various attributes. The need for Very High reliability stems from past financial losses due to inadequate motion detection. Database expansion without cleanup, as does queue handling for HomeKit communication, increases complexity. High execution time, storage constraints, and platform security issues demand additional resources. Although turnaround time is Low, skill deficiencies across personnel and inexperience with tools amplify the need for more staff. These challenges require a larger team to manage complexity, ensure reliability, and effectively meet project deadlines.

# **Part 3: Ideal Conditions Scenario**

***Cost Drivers Adjustments***

|  |  |  |
| --- | --- | --- |
| **Project Factor** | **Value** | **Explanation** |
| **Product Attributes** |  |  |
| Required Reliability | VL | Ping developer automatically to fix issues. |
| Database Size | L | Tiny database footprint. |
| Product Complexity | L | Integrating with sensors will not require complex predicates, and there is no need for heavy code nesting. |
| **Computer Attributes** |  |  |
| Execution Time Constraint | N | Same as before |
| Main Storage Constraint | N | Same as before |
| Platform Volatility | L | Same as before |
| Computer Turnaround Time | L | Same as before |
| **Personnel Attributes** |  |  |
| Analyst Capability | VH | Personnel is highly capable. |
| Applications Experience | VH |  |
| Programmer Capability | VH |  |
| Platform Experience | H |  |
| Programming Language and Tool Experience | H |  |
| **Project Attributes** |  |  |
| Modern Programming Practices | VH | Best in class project. |
| Use of Software Tools | VH |  |
| Required Development Schedule | VH |  |
| **New** |  |  |
| Required reusability | N | No reusable components are required. |
| Documentation match to life-cycle needs | XH | Great specifications available. |
| Personnel continuity | XH | Same engineer continously. |
| Multisite development | VL | 100% Local. |

***Estimation Report***

A screenshot of a computer program

Description automatically generated The staffing estimate drops to 1.155 due to critical factors across various project attributes. Very Low Required Reliability is achieved through automated issue alerts, minimizing manual intervention. Low Database Size and Product Complexity simplify tasks, reducing personnel needs. High Analyst, Programmer, Applications Experience, proficient Platform Experience, and Modern Programming Practices optimize team efficiency. High ratings for Use of Software Tools, Required Development Schedule, and Personnel Continuity further streamline operations. Extra High Documentation match and Personnel continuity ensure seamless knowledge transfer. Lastly, Very Low Multisite development indicates a localized project setup. These factors collectively drive down the staffing estimate, reflecting optimized processes and skilled personnel.

# **References**

|  |  |
| --- | --- |
| [1] | J. F. Nusairat, Rust for the IoT, Berkeley, CA: Apress L. P., 2020. |
| [2] | N. Aeronautics and S. Administration, *STRS COCOMO Calculation,* 2021. |