AndroidMonke Productivity

Configuration Management Plan for the Development of NO Monkeying Around!

Table of Contents

1. Introduction

1.1. Purpose

This configuration management plan (CMP) shall set forth the practices and standards that will guide the present and future development and maintenance of NO Monkeying Around! (NMA!). This configuration management plan shall be used to inform any interested parties in the development practices of AndroidMonke Productivity pertaining to NMA!

1.2. Scope

The scope of this CMP is solely related to the development of NMA!, be it in documentation, structure, code, or anything else that relates to the development of NMA!.

1.3. Key Terms

No Monkeying Around (NMA!): The title of the application under development

1.4. References

NMA! SRS Document:
https://docs.google.com/document/d/13r-cDMsu55Cuw4JDIFDCzHrUZTbVyBgC

2. SCM Management

2.1 Organization

The creation of NMA! is a product of AndriodMonke Productivity. AndroidMonke Productivity is composed of four students at UMaine's COS420 course, which is Software Engineering. We four are responsible for the development and maintenance of NMA!, along with the adherence to this CMP.

2.2 Responsibilities

- Product Owner
  - Purpose: delegate responsibilities, informs the team of deliverables
  - Membership: one member
  - Period of effectivity: two weeks, after wish the team shall have a meeting wherein roles are shifted, discussed, and decided
  - Scope of authority: superior to all
  - Operational procedures: manage scrum and product backlog, create documents
- Scrum Master
  - Purpose: enforce scrum protocols, ensure team success
  - Membership: one member
  - Period of effectivity: two weeks, after wish the team shall have a meeting wherein roles are shifted, discussed, and decided
  - Scope of authority: inferior to Product Owner, equivalent to Development Team
  - Operational procedures: manage sprint backlog
- Development Team
  - Purpose: handle the general responsibilities, deliver work through the sprint
  - Membership: two members

- Period of effectivity: two weeks, after which the team shall have a meeting wherein roles are shifted, discussed, and decided
- Scope of authority: inferior to Product Owner, equivalent to Scrum Master
- Operational procedures: manage development of NMA! and create documents

2.3 Application Policies, Directives, Procedures

We are not working under any explicit constraint. We have two forms of implicit constraints. Firstly, those outlined by the preset deliverables from COS420, which are due approximately every two weeks, wherein certain documents are required to be complete. Secondly, constraints we set on each other to have certain features, or artworks, and so on, due by a certain date or time.

3. SCM Activities

3.1 Configuration Identification

3.1.1 Acquisition of Configuration Items

Configuration Items (CIs) will be identified for NMA! through the following means:

- Review of system requirements and functionality
- User Stories
- Focus group feedback and potential customer discussions
- Changing functionality of the application

These means as well as the continued development of application source code will be used to determine different configuration items. These items will also be updated/changed during future sprint cycles.

3.1.2 Anatomy of Configuration Items

The naming scheme of the CIs shall be concise and informative so as to allow for the easy retrieval and understanding of what each CI will entail. The rules governing the naming scheme of CIs shall be as follows:

- All names shall be simple and highly descriptive
- All names must include a numerical identifier

All CIs must also carry with them:

- A description of the CI
- A description of the reasoning behind the inclusion of the CI
- Optionally a list of resources to aid in the understanding of the CI

3.2 Configuration Control

3.2.1 Change Requests

Changes to the application must be requested either in a scrum meeting, or by opening an issue in the Kanban board/Github. Change requests must contain the following items in order to maintain a high level of organization:

- Change request title
- A description of the desired changes, including how the changes will be implemented, and how the changes will affect the application functionality/user experience

- A description of why the changes are necessary
- A preliminary assessment of how long the changes will take to perform, along with a deadline for change implementation

3.2.2 Evaluating Changes

Changes are to be discussed as per 3.2.1 and evaluated as a group at our Scrum meetings wherein the group will hold a vote on the efficacy of the change. If the majority of the groups decides the change is positive or necessary, the change shall be implemented as described in 3.2.3

3.2.3 Implementing Changes

Changes shall be implemented by the dev team after and only after a vote has been held within a scrum meeting. These charges are to be implemented via a pull request from the development team, upon which they shall be reviewed by one or more members of the dev team. If the change is resolved, it shall be merged into an upstream branch and the issue is now closed. This chain follows similarly for all non-code based changes, where they shall be evaluated, implemented, and submitted.

3.3 Configuration Status Accounting

Configuration Status will be recorded/tracked in the following ways:

- Version control system commits
- Progress of CI implementation
  - This will be reported by the respective developer of each CI
- Time spent working on CI implementation
  - This again will be reported by each respective developer

Access to the status accounting reports will only be available to management/project leaders and the development team. End users will not need access to this information.

3.4 Configuration Evaluation and Reviews

3.4.1 Audits of Configuration Items

An audit of any CI that is to be implemented shall take place. Within this audit, one or more members must ensure that the CI meets the standards of the mainline of NMA!. The specifics of each audit may vary, though each will address whether the CI makes broad changes to the basis of NMA!, and require code pulling from the GitHub repository.

3.4.2 Audit Requirements

- Objectives:
  - Ensure the implementation of any CIs lead to positive chgange within NMA! be it through bug fixes or the moving forward of the application via version control.
- Schedule:

    - All audits are to be accomplished without a schedule, though large changes to the application are to be brought to the attention of AndroidMonke Productivity via the weekly scrum meetings.
- Procedures:
    - The procedure of audits is described above in section 3.4.1.
- Participants:
    - The participants of an audit are to be one or more members of AndroidMonke Productivity.
- Approval criteria:
    - The audit is considered to be approved when all changes are merged with the mainline of NMA!.

3.5 Interface Control

Configuration items with impacts outside the scope of the immediate development environment shall be closely monitored by the development team and management staff. Major changes to out of scope CIs will require approval by management, and before the changes are comitted, a high level analysis must be conducted about the potential impact to the end user and customers.

3.6 Subcontractor/Vendor Control

Any subcontracting that may occur with regards to development of CIs must go through a similar analysis process as described in the interface control section above. The product owner and management will work closely with the respective subcontractor to ensure major changes do not have a detrimental impact to the overall product and user experience.

3.7 Release Management and Delivery

For every release, the following policies must be followed:

- The development team will determine when new releases will occur. Management will have an opportunity to suggest release dates, and the development team will give feedback about if the suggested release dates are realistic.
- Releases will only occur after all CI changes are audited and determined to not have detrimental impact on the product.
- New releases of the application will be available to the user through the Google Play store.

For major releases:

- In addition to the policies listed above, major releases will go through an additional audit process in order to ensure it will not have detrimental impact on the user experience as well as the backend database system. Users will also be able to see patch notes for the release in the Google Play Store.

For minor releases and patch releases:

- The policies laid out above will be followed for minor and patch releases. The development team will also determine whether patch notes will be released along with the application update.

4. SCM Schedules

4.1 Sequence and Coordination of SCM activities

The sequence in which all SCM changes are to be implemented is as follows:

1. A formal change request is made to the group
2. A developer is tasked with the implementation of, or the delegation of the implementation, of the change.
3. The development team is tasked with creating a new branch and implementing the change.
4. A pull request is made to merge the new branch and the main branch.
5. The audit process occurs by one or more members.
6. The development team merges the new branch and the main branch.

4.2 Relationship of Key SCM Activities to Project Milestones or Events

- Configuration Items shall be identified before development begins, typically on the first 1 to 3 days of a sprint.
- Implementation of these changes shall occur continuously throughout the sprint without clear deadlines.
- The audit process shall begin on day 12 of the sprint and be finished by day 13 of the sprint, though may occasionally continue into day 14 of the sprint

4.3 Schedule either as absolute dates, relative to SCM or project milestones or as sequence of events.

For each sprint, the timeline of SCM activities will follow the format detailed below:

- Day 1-3:
  - The team will outline the user stories and application functionalities to be developed.
  - Delegation of different SCM activities will occur.
- Day 4-6:
  - The team will meet and discuss SCM activity progress.
- Day 7-10:
  - A preliminary audit will be conducted about the impacts of SCM activities on the product.
  - Anywhere from 50-60% of the work should be completed by this date.
- Day 11-14:

    - The team will meet again to perform the final audit of the SCM activities and prepare documentation and source code for release.
- Day 14:
    - Release will be implemented.

5. SCM Resources

5.1 Identifies environment, infrastructure, software tools, techniques, equipment, personnel, and training.

- Environment
    - The two programming languages we are using are XML and Java. XML handles the display and configuration of the images. Java handles the Android back-end and the database creation and access, which uses SQLite. The user has no access to SQLite manipulation, though the user may interact with the XML configurations through the application UI.
- Infrastructure
    - Each project demands different infrastructure requirements, within which we do not specify any infrastructure or design. We adhere to this CMP whenever, wherever, and as much as possible.
- Software Tools
    - GitHub
    - AndroidStudio
- Techniques
    - We are utilizing Github and an Agile development methodology. We do not hold any coding techniques as standard. Proper utilization of Github requires a README, pushing documentation and code elements to a shared repository, and the pulling of code from our shared repositories. Agile demands we use certain roles (as defined in 2.2) and follow practices outlined throughout this document.
- Equipment
    - The current equipment being used are as follows:
        - AndroidStudio IDE
        - Declan's Android phone
        - Emulation
        - Local server
    - Upon rollout of NMA!, we would need to acquire:
        - Cloud-based database
- Personnel
    - There are presently five members of AndroidMonke Productivity
        - Sean Staton
        - Declan Brinn

        - Caiden Emerson
        - Matthew Brown
        - Travis Nickerson
- Training
    - There are no training protocols as of yet for new members of AndroidMonke Prouctivity as we are not currently seeking the addition of members to the team. We do not seek to add another group member to the project before the project's end.
    - For the existing members of AndroidMonke Productivity, we are presently taking a hands-on approach to training ourselves wherein we gather outside materials, adapt them to our desired outcomes, and share our results with our team members.

5.2 Key factors for infrastructure

- Functionality
    - We will adhere to the SRS document for the functionality of NMA! wherein the features and layout of NMA! is described. Strict adherence to the SRS is expected, though these functionalities are not fully expected to be completed before the May 1st deadline.
- Performance
    - There are presently no key performance factors on the development of NMA!.
- Safety
    - There are presently no key safety factors on the development of NMA!.
- Security
    - Security is to be held via the hashing of user's passwords and general database security to be held in high regard due to the dealing with personal information such as names, ages, passwords. This information is to be securely held within a database via a database management system.
- Availability
    - Due to NMA! currently being in development without a clear rollout date, there is no expected availability benchmark set forth by AndroidMonke Productivity.
- Space Requirements
    - There are presently no key storage factors for the implementation and development of NMA!.
- Equipment
    - In order to use NMA! the end user must use the Android operating system with a minimum API level of 23.
- Costs
    - There are presently no key cost factors on the development of NMA!.
- Time Constraints

    - The development of NMA! is limited to UMaine's Spring Semester of 2022, with the final sprint ending on May 1st, 2022. There are presently no plans to continue development past this date, though this is subject to change.

5.3 Tools for Each Activity

- Github
    - Code storage
    - Remote code access
    - Version control
    - Documentation
- AndroidStudio
    - Application Development
    - Emulation
- Google Drive
    - Document Storage

6. SCM Plan Maintenance

6.1 Plan Monitoring

The monitoring of this plan primarily falls to the Product Owner. However, it is the combined responsibility of the entirety of AndroidMonke Productivity to maintain

6.2 Update Frequency

We shall inspect this CMP weekly to verify if updates are necessary. Mandatory updates to this CMP will be completed biweekly.

6.3 Plan Changes: Validation and Approval

It is the responsibility of the Scrum Master to ensure the validation and approval of changes to this CMP upon updates done by either the Project Manager, Scrum Master, or Development Team.

6.4 Plan Changes: Formation and Communication

Changes to this CMP are to be primarily completed by the Product Owner, though can be completed by any of the team members. Communication of changes to this CMP are to occur at scrum meetings via the Scrum Master when updates to the CMP are discussed.

6.5 Plan Changes

Changes to the plan are to be completed as described in 6.2, 6.3, and 6.4.