B.U.B.O.L.O. Software Configuration Management Plan

CS673 Clone Productions
5 February 2014

1.0 Introduction - The SCM plan guides the project team on how configuration management will be executed on the project team. It defines management of the process, what defines a configuration item and the process of executing a change to any baseline configuration item.

2.0 SCM Management

2.1 Organization - This team is comprised of a simple organization. One project manager coordinates tasks across three team leaders and those leaders then coordinates tasks within the team itself. The structure is shown in Figure 2.1 - 1.

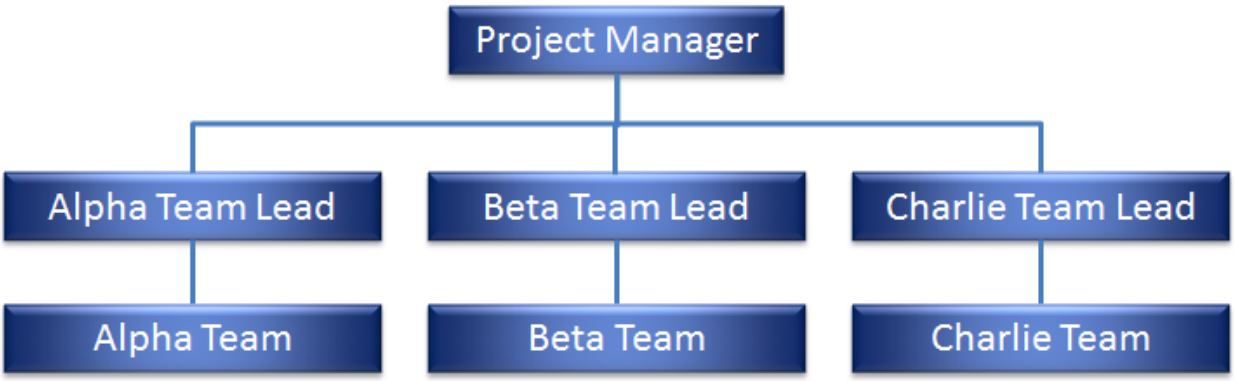

Figure 2.1 - 1

2.2 SCM Responsibilities - The Configuration Manager is responsible for ensuring that branches occur on the same version of the item and ensures that merging occurs in such a manner that all changes are incorporated as approved. This is executed through the change process outlined in paragraph 4.0.

2.3 Applicable Policies, Directives & Procedures - Change approval and implementation will occur following the processes outlined in this document and ensure compliance with the Software Quality Assurance Plan for testing and managing defects.

2.4 Management of the SCM Process - While the configuration manager has the

responsibility of ensuring these processes are executed properly, each individual on the project must ensure that these processes are executed in coordination with the configuration manager.

3.0 SCM Activities

3.1 Configuration Identification - Baselines will be identified utilizing a three digit number convention such as 0.0.0. The first digit will represent a finished release version, the second digit will represent sprint progression builds, and the final version will represent intermediate configurations between sprint builds.

3.2 Identifying Configuration Items - Configuration Items (CI) will include any documentation after initial release of the document and code files once they are merged into a subteam's branch. It will also include game assets, such as sound and art files that are used in the presentation of the game.

3.3 Naming Configuration Items - CI's will either be identified by version number for code or rev letter for documentation.

3.4 Acquiring Configuration Items - Configuration Items will presented within the subteam where they are drafted for review prior to being merged into the subteam's branch. Configuration items being merged from a team's remote branch into the production or master branch will be reviewed according to the SQAP. Once the proposed CI is added to the program baseline, it will then become a managed CI.

4.0 Configuration Control

4.1 Requesting Changes - Changes to configuration items are expected throughout the duration of the project. Each team member will be responsible for creating and updating configuration files -- including code artifacts, assets, and documents -- to further the completion of the tasks outlined in the "design document." A team member may make changes to his or her local branch at will, so long as all changes are documented and a summary of the changes made with is recorded with each commit. Team members may merge their changes with the team's remote branch with the permission of their Team Leader. Changes to the master branch will be made in accordance with paragraph 7.0. Changes to the production branch may be requested by any team leader, but must be approved by all teams and implemented by the Configuration Manager.

4.2 Evaluating Changes - Each team leader is responsible for ensuring that any

changes made to his or her team's remote branch have passed the testing procedures outlined in the SQAP and are suitable for incorporation into the master branch. Said changes should be documented, and a history of all changes should be available to all project members. Similarly, the Configuration Manager is responsible for ensuring that any changes that will be incorporated into the production or master branch follow the same guidelines.

4.3 Change Approval/Disapproval - If a Team Leader approves a change to be made by one of his or her team members, said change will be merged into the team's remote branch. If the Team Leader disapproves of the change, the team member requesting the change will be notified and informed of the reason for the disapproval. Should a team member wish to contest this disapproval, he or she should first contact his or her team leader to discuss the decision. If the team member is still not satisfied following a discussion with his or her team leader, or is unable to have such a discussion, he or she may contact the Project Manager, who will determine whether the decision was justified and take the appropriate action at his or her discretion.

If the Configuration Manager approves of a change made by a Team Leader, said change will be merged into the production branch. If the Configuration Manager disapproves of the change, the Team Leader requesting the change will be notified and informed of the reason for disapproval. This decision may be contested in a similar manner as performed with changes to remote team branches: the Team Leader should contact the Configuration Manager to discuss the decision, and if the disagreement cannot be resolved, the Team Leader should contact the Project Manager, who will take the appropriate action at his or her discretion.

4.4 Implementing Changes - Each Team Leader is responsible for ensuring that approved changes are merged successfully with his or her remote branch, which should include both a summary of any major changes and a complete history of all changes made to each configuration item. The Configuration Manager is responsible for ensuring that changes are merged into the production and master branches, and must follow the same guidelines as Team Leaders regarding the change history of each configuration item.

5.0 Configuration Status Accounting - The Configuration Manager is responsible for periodically evaluating the configuration status of the project. This includes the inspection of each remote branch to ensure that an accurate change history is being recorded for all configuration items, and that proper configuration control procedures are being followed. These evaluations should be performed on each remote branch at least once before the end of each sprint, and once on the master branch before each release. Additional inspections should be made at the discretion of the Configuration Manager. If any

deviations from the configuration control procedures are identified, the Configuration Manager is responsible for bringing these issues to the attention of the appropriate Team Leader(s) and the Project Manager, and for taking action to bring about the correction of any discrepancies.

6.0 Interface Control - Because the project will be divided into several modular components, the interfaces between these components must be clearly defined and maintained to ensure proper functioning of the overall program. The outline for the interface control scheme will be included in our “design document,” and will be constructed with input from all members of the team. This document will contain specifications for the components of the project, including:

- the scope of the responsibilities for each component, in the context of the overall program,
- the interfaces by which the component will interact with other components, and
- the team responsible for the creation of code artifacts which will fulfill these responsibilities and adhere to the specified interface guidelines.

The leader of each team will be responsible for ensuring that his or her component fulfills its responsibilities and adheres to the interface guidelines. In the event that responsibilities or interfaces be added, removed, or transferred between components, such changes should be approved by all affected teams before being added to the “design document.”

7.0 Release Management and Delivery - Releases will be scheduled by Professor Czic. In order to prepare for these releases, Sprints will be scheduled in advance of each release, which will allow teams time to prepare code artifacts to fulfill the requirements set forth in the “design document.” Team leaders are responsible for ensuring that the remote branches associated with their teams are updated at the end of each sprint, at least 3 days in advance of the release deadline. Changes to any configuration items made after this time will not be included to the impending release without approval from all teams.

Such updates should include both a summary of major changes made to any affected components since the last release and a full record of all minor changes. Individual team members are responsible for submitting descriptions of said minor changes along with any updates they make to their team’s remote branch. These changes include modifications to any documents, including the creation of new documents or assets.

Changes made to each team’s remote branch will be merged into the production branch by the Configuration Manager. This branch will undergo testing in accordance with the SQAP. If this testing indicates that the software is suitable for release, a new baseline

will be recorded, and the code will be merged with the Master branch. This baseline will be given a new version number, and will include a list of changes made since the last baseline. The source code will then be compiled into binary files, which will be staged for distribution over the Internet using Sourceforge.

8.0 SCM Plan Maintenance

- This plan will be reviewed monthly and updated for any changes necessary. Any changes made to this document will take effect once the final updates are released. Team members may request changes to this plan in advance of scheduled updates by contacting the Project Manager.