Doug Richardson, Jon von Kampen, Chris Fibich, and Nick Watkins.

Proposed list of tasks and priorities

Here are the proposed priorities as our team understands them. This document is prior to the work breakdown structure. It is assumed that any task in the WBS will have the same priority as its “parent.” This document did not include early conceptualization tasks (which are assumed to be required).

This document will be using a derivative of the MoSCoW system (the only difference is I use "would have" and not "won't have," indicating a low-priority desire to actually put it in rather than ignore it completely).

For more information, please refer to http://en.wikipedia.org/wiki/MoSCoW\_Method

This document will be revised to reflect the work-breakdown structure, but as stated above, all “children” tasks can be assumed to have equal priority to that of their “parent.”

THIS DOCUMENT IS SUBJECT TO CHANGE (with agreement from the team and the client/stakeholders).

**Nomenclature of priorities used**.

M: MUST have. These are the things that are ABSOLUTELY REQUIRED for the system to be considered complete. The ABSOLUTE MINIMUM requirements are for EVERY M task to be accomplished.

S: SHOULD have. These are high priority tasks that should be accomplished if at all possible. Often almost as important as MUST tasks, but aren't as time-critical or can be worked around if necessary. These are what we focus on once we have our M's well established (or if one of them is so easy to do we can get it out of the way). Basically, they are "very important, but not required in the strictest sense."

C: COULD have. These are medium priority tasks. Things that would be nice for the system to have, and would definitely help. However, these take a back seat to the first two (ESPECIALLY M's) unless it can be done with a minimum of interference to the core project.

W: WOULD have. These are the lowest priority tasks. These are the things that we'll do last if we get ahead of ourselves (even if it seems easy, the effort is deemed not worth diverting energy from the previous three).

These CAN be upgraded/downgraded with agreement from the stakeholders, so they aren't set in stone (although we should work as though they are until circumstances dictate otherwise).

**Feature list** (this is just a "scattershot brainstorming" of the list, it's in no particular order).

Note: The tasks that are part of the security sub-project are written in red.

**M TASKS**

* Project must contain usable database of CWEs
* Project must contain a means to populate the database with CWE data (currently stored in XML format)
* Project must contain a web interface for end users to browse CWEs (more usable then the current one).
* Project must have a means of connecting the web application to the database
* Project must have a means of searching CWE's other than just the ID number
* Project must encode any CWE sample code, such that it will not execute if displayed in a web browser.
* Project must encode any user input displayed on-page (*e.g.* search terms) to prevent cross-site scripting attacks.
* Project must encode any user input in searches to prevent database injection attacks.
* Project must verify that search filters originating from static values (*e.g.* dropdown list items) have not been modified to contain unexpected data.

**S TASKS**

* Project should have a means of auto-updating the database with the latest CWE information (assuming consistent schema).
* Project should have a searching mechanism with full-text-body search functions
* Project should have a prioritization algorithm for full text body searches (example: XXX in the title trumps XXX in the description).
* Project should have filtration mechanisms in searching (filter by language used, etc.).
* Project should have a means of Text Mining common terms ("a," "the," etc.) to avoid throwing the results off.
* Project should have a "Back-trace log" of path taken to facilitate navigation and "mapping" the relationships of CWE's researched.
* Project should have a means of input validation for security purposes
* Project should use expert-reviewed, popular security libraries and concepts in preference to devising original functions.
* Project should undergo security testing by manual and/or automated fuzzing prior to final delivery.
* Project should undergo security code review prior to final delivery.
* If admin mode is implemented, project should authenticate admin actions.
* Project should not publicly display error messages that reveal internal architecture (language, framework, etc.).

**C tasks**

* Project could have the CWEs stored as a graph for visual/mapping purposes.
* Project could have a means of visually showing the relationships between CWEs (Research/Development CWEs)
* Project could have a shortest path algorithm between two given nodes, to calculate the smallest possible distance
* Project could have a means of altering XML dumper to conform to a major revision
* Project could have an admin mode for "manually" updating the site without having to physically dig into the code.
* Project could have a means to redirect to the old CWE site from the page presented (it should not need to, but the option can exist).
* Project could have a means to show differences between CWE versions (if we have multiple databases for whatever reason, which version says what, etc.).
* Project could have a means of displaying the current version of the database
* Project could provision individual user accounts for application servers, database servers, and other project-related executables.
* Project could provision executable accounts with the minimum necessary system privileges.
* If admin mode is implemented, project could include pseudo-random tokens in each admin form to prevent cross-site request forgery attacks.

**W TASKS**

* If time permits, project would have a means of visually showing complete or partial visual maps between the CWEs
* If time permits, project would have a means of visually representing CWEs by category
* If time permits, project would undergo security testing by manual and/or automated fuzzing prior to each minor customer deliverable.
* If time permits, project would undergo security code review prior to each minor customer deliverable.

**Developer priorities**

This project will prioritize usability in the form of how easy the application is to search and navigate. Performance (it's a big database) and security (it's a public website) are concerns as well. Development priorities will focus on quality development of the M and S systems above quantity of tasks completed.