

HYDERABAD SOFTWARE EXPORTERS ASSOCIATION
Organizes
STUDENTS COMPETITION ON
SOFTWARE DESIGN
January 2007

Topic : Design for Software Code Reviews

Supported by:



CODE REVIEWS:

A Code review involves one or more software engineers examining source code they didn't write and providing feedback to the authors, both negative and positive. Code is reviewed for syntax, standards defined, readability and maintainability (code organization, comments etc). Typically, the reviewers will have a standard checklist as a guide for finding common mistakes and to validate the code against company's coding standards.

Code reviews ensure standards, save time and cost, reduce number of bugs early in the development life cycle and serve as a process improvement mechanism.

WHY REVIEW CODE?

It is human nature that one cannot adequately find mistakes in own work. Authors need editors to catch mistakes. All code reviews have the following goals:

- Defect free, well documented software
- Software that complies with company's coding standards
- Teaching and sharing knowledge between software engineers
- Ensure maintainability

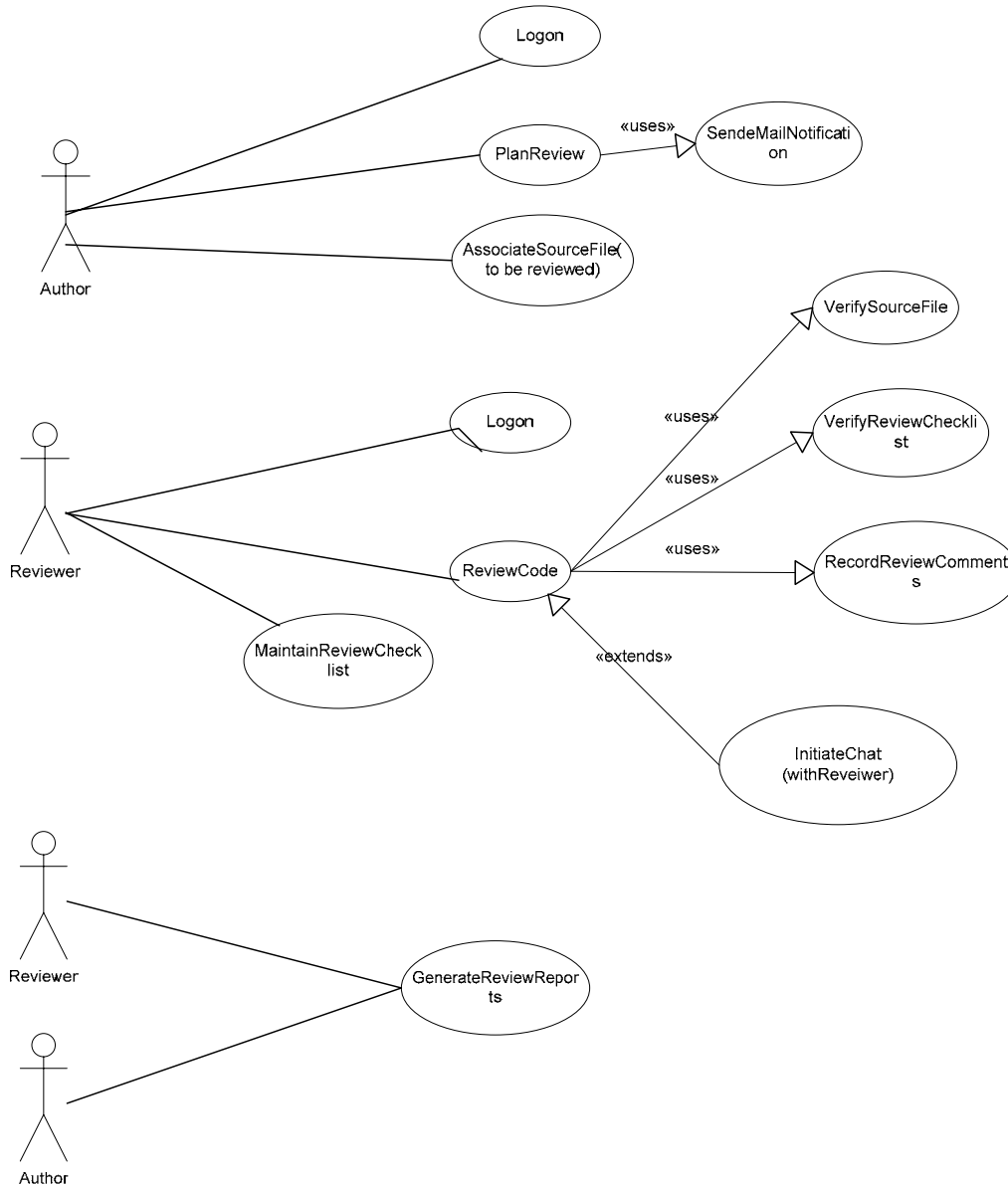
PROBLEM DEFINITION:

Design a product for code reviews with the following guidelines. Any assumptions made beyond the following guidelines need to be documented:

- allow for at least two reviewers simultaneously
- reviewers may be remote and use web to access the code
- reviews are planned by author by intimating reviewers in advance of the planned date and time of the review
- Usability of system - Keep it simple
- System should be web-enabled
- Build suitable security features for access of code and ensure control on code under review
- Provide for interaction between author and reviewers during review to seek clarifications etc.(tip:chat, webex help in online interactions). Also facilitate document sharing (typically designs to validate code, standards etc.) between the author and reviewers during the review.
- Create a repository to log reviewer comments and action taken by author, both with date and time stamp to analyze time taken for actions. This repository will be used in future to analyze comments/ bugs, actions taken, time taken, and for updating the checklist for prevention of defects
- Provide for storing the Code Review checklist that will be used by reviewers during the review. Ensure proper permissions / version control for checklists so that only authorized personnel can make changes / store the checklist. All others except authorized personnel will have read access only to the checklist. Ensure proper version control on the checklists with appropriate information (tip: name of person who modified the checklist with date and time stamp)
- Define appropriate metrics (at least three but more desired) to measure quality of code and efficiency of Peer Review Process.
- Provide for online reporting and graphical displays of metrics over a period of time asked by the user

- Suggest appropriate technology that works independent of system configuration that the author / reviewer may use
- Design should be scalable to accommodate any other reviews Fagan Inspection in future

The below use case diagrams are only illustrative and for your understanding only. You can add or modify or delete the use cases to meet the problem statement.



Hint:

1. *DO not reinvent the wheel. You may refer to the commercial off the shelf products available on the net for some ideas and improve on them.*
2. *Please go through online help material to get more insights into concepts and ideas of design. Some of the websites you can search are*

- a. Campusconnect.infosys.com (id: demo@infosys.com ; Passwd: infosys)
- b. www.google.com (very commonly used search portal)
- c. en.wikipedia.org (free online encyclopedia)
- d. www.usabilitymatters.org

SELECTION CRITERIA FOR BEST DESIGN:

- Completeness/Quality
- Simplicity
- Innovation / Originality
- Features/substance
- Feasibility
- Scalability
- Technology selection / Architecture

Attractive Prizes for the finalists!!

KEY MILESTONE DATES

- 01st Nov 2006 . Announcement of the Competition
- 08th Nov 2006 . Issue clarification program-1 on Mana TV
11:00 am to Noon. Contact Phone: 99490 92452
- 14th Nov 2006 . Last day for submission of Abstract
- 20th Nov 2006 . Issue clarification program - 2 on Mana TV
- 05th Dec 2006 . Last day for submission of Designs
- 16th Dec 2006 . Announce short-listed designs
- 05th Jan 2007 . Presentation of prototypes to Panel of Judges
- 09th Jan 2007 . Prize distribution to winners during GITEX.

Contact info:

For information or clarification, please send a mail to scsd_hysea@infosys.com

ACKNOWLEDGEMENTS (in alphabetical order):

HYSEA acknowledges active support of:

Gudipudi Sudheer
Kanakaiiah
Narsimharao M
Prasad Sivalanka
Srinagesh Chatarajupalli
Suren Poruri

COMPETITION INFORMATION AND GUIDELINES

1. Each team should comprise to a maximum of three (3) members only
2. Each college can have maximum of two (2) teams participating in the competition
3. Five (5) teams will be short-listed for finals.
4. The finalists would work on prototype of the design that they have suggested.
5. One person can be part of only one team.
6. Finalists will be notified individually and to their college.

7. Winners of the competition will be announced during the 2 day GITEX event in Hyderabad.
8. Any document that does not follow the format mentioned will automatically be disqualified for the competition
9. The design document should be sent as word document or PDF file.
10. All submissions become property of HYSEA.

<<Title of the Paper>>

Author(s) Names:

Some general guidelines:

. Focus your paper on the core idea(s) . at least 50% of the paper should be devoted to the topic at hand This is not a tutorial; do not explain background, introduction or basic concepts at length;

The average size of the paper should be 8-10 pages of MS Word

. Do not copy abstract to the conclusion section. Abstract should have what the paper is focused on; while conclusion should include specific, measurable outcomes of using the said approach

. Use tables, diagrams and pictures. Refer to these in the text. In case there is no discussion of these exhibits in your text, remove the exhibit

. Do not use adjectives. Qualify statements with metrics (e.g. large project vs 350 man-month project)

. Acknowledge source of information (e.g. data from Gartner paper xxx pages 5-7 or connect to one of the references)

. Use standard terminology

. Use simple language and short sentences.

. Use the third person (Avoid usage of .I. and .You.)

. Avoid jargon. When you use technical terms/ jargon/ acronyms/ abbreviations, please explain them in brief, the first time they appear in the paper.

. Adhere to the sequence of the paper as suggested in the sample below

Introduction

(Heading 1 . Arial, 12 size, bold, paragraph setting before 0, after 6 pts, line spacing single)

It contains the broad overview of the paper (no details); set the context for the paper.

Main Body of the Paper

It should include headings, sub-headings, illustrations (to demonstrate results), More than 90% of the paper should be main body.

The main body of the paper should cover the following:

. Problem definition

. Proposed solution

. Architecture

. Environment

. Performance Attributes

. Functional overview / Flow diagram

. Database relationship / ER diagram

. Constraints / Exceptions

(Heading 2 . Arial, 11, bold, paragraph setting before 0, after 3 pts, line spacing single)

(Heading 3 . Arial, 10, bold and italics, paragraph setting before 0, after 3 pts, line spacing single)

Please note that the figure number should be referred as : Figure 1, Figure 2, Figure 3 etc instead of .refer figure below. or .Figure shown above..

Conclusion

This is the final section of the main body of the manuscript and it is mandatory. This section must be well connected to the abstract. It must not be a restatement of

the abstract. It must draw points from the discussion presented in the body of the manuscript. It must be written in a brief and logical manner.

References

All sources of information (artwork or otherwise) including full name, title (if applicable), address and phone number/Web page must be listed.

At the end of the article, references must be arranged in alphabetical order of authors. surnames and chronologically for each author. The author's surname is placed first, followed by the year of publication in parentheses. For more details, refer MLA Handbook for Writing Research Papers

Sample:

[1] A.B. Smith, C.D. Jones, and E.F. Roberts, .Article Title., Journal, Publisher, Location, Date, pp. 1-10.

[2] Jones, C.D., A.B. Smith, and E.F. Roberts, Book Title, Publisher, Location, Date.



Abstract for HYSEA Students Competition

Name of the team

(how do you want your team to be called)

Authors Name(s)

(indicate the primary author within bracket)

Name of the College and address

Tel number and e-mail of the primary author

Abstract Title

Body of the abstract (not exceeding 250 words)

Please submit the abstract to scsd_hysea@infosys.com

Author Declaration Form

**Name of the team
(how do you want your team to team called)**

Name of Primary Author

Name(s) of other Author(s)

Class / Discipline

Name of the College

Address

City / State

PIN/Zip

Phone / mobile

Email

Paper Title

I hereby certify the following:

- . The content / structure that forms part of the above paper is original and has not been copied from an existing source of content
- . The paper is not a reconstruction of the whole or part of an existing source of content.
- . The sentences and key phrases in the paper have not been copied from an existing source of content.
- . All the reference material used in the development of the paper is listed as an annexure to the submission for the paper.
- . Papers / designs submitted will be the property of HYSEA.

Signature:

Date:



<<Title>>

Design Document

Name of the team:

Name of the Author(s):

Name of the college:

REVISION CONTROL CHART

© Copyright 2006; this document is the property of Hysea

REVISION CONTROL CHART					
Revision	Date	Changes	Affected pages	Prepared by & date	Reviewed by & date
0.1	24-10-2006	Initial version	All		

Table of Contents

The sections mentioned are indicative and any additional topics can be added at appropriate sections.

1. Introduction
2. Definitions and Acronyms
3. Requirements
 - 3.1 Problem Definition
 - 3.2 User Interface requirements
 - 3.3 Performance requirements
 - 3.4 Constraints
4. Solution Proposal
 - 4.1 Proposed Solution
 - 4.2 Architecture
 - 4.3 Environment
 - 4.4 Performance Attributes
 - 4.5 Functional overview / Flow diagram
 - 4.6 Database relationship / ER diagram
 - 4.7 Constraints / Exceptions
5. Conclusion
6. References

Glossary

Usability - Ease with which a user can learn to operate, prepares inputs for, and interprets outputs of a system or component. The system with good usability will be treated as 'user friendly' system.

Security - The ability of a system to manage, protects, and distributes sensitive information. Security of the system should prevent the unauthorized access to the system and its data.

Maintainability - The ease with which a software system or component can be modified to correct faults, improve performance, or other attributes, or adapt to a changed environment.

Coding Standards - Programming or coding conventions for writing source code in a certain programming language. Coding standards are useful for ease of maintenance; ease of understanding and to get good performance.

Typically, in software industry each company may have it's own coding standards for each programming language. These standards will be followed by all the developers across the organization in various projects.

Webex - It's a kind of tool to support online meetings, web conferencing and video conferencing services

Metrics - A quantitative measure of the degree to which a system, component, or process possesses a given attribute. Software metric is a measure of some property of a piece of software or its specification. e.g number of defects per lines of code;

Annexure: Students Kit

Given below are the templates (with some example entries) for the documents related to a sample project. These are just guidelines only. This document acts like reference copy only.

Requirements Specification (RS)

Following is a template for the RS document. Some example requirements are entered in it to show how to use the template. Make sure that you enter even the smallest/most trivial requirements also.

No.	Requirement	Essential or Desirable	Description of the Requirement	Remarks
RS1	The system should have a login	Essential	A login box should appear when the system is invoked.	The logins are assigned by the mail-admin
RS2	The system should have help screens	Essential	Help about the various features of the system should be provided in sufficient detail in a Q&A format.	For example help should be provided for : How to create a defect. How to modify a defect etc.
RS3	The system should 'lock' the login id if wrong password is entered 3 times in a row	Desirable	This feature will improve the robustness of the application	Since the application is going to be used only by the employees of the organization, this feature is not essential. However, if time is there, this will be implemented.
RS4				

Database Fields Specification

Following tables may be created to serve the application:

User table – Contains details about the users of the system (their name, role, access privileges etc)

Defects table – Contains details about the defects created by the users of the system (defect id, creator’s id, status of the defect, owner id, date/time of creation/last modification etc)

User table is taken as the example for specifying the field details. Some example entries are filled in.

No.	Field Name	Range of valid values for the field	Remarks
1	User ID	To be defined by the team.	This is the key field of the database as it is unique for a user. This will also serve as the login for the system.
2	Name	Up to 30 characters in length.	Special characters like underscore are not allowed.
3	Access Details	Application Admin / User	
4	Email ID		Should be a valid email ID.

High Level/Detailed Design (HLD/DD)

Overview of the system

Provide a high-level block diagram depicting where the database will be located, where the application will run etc.

Detailed Design

Split the system into its design components. In this case, the components would be user-verification, defect creation, mail notification, component maintenance, report generation etc. For each of the components, provide information in the following format. User-verification component is taken as the example.

Component one

User-verification

Purpose

This component will verify if the user who is trying to access the system is a valid user.

Component two

Component three....and so on....