

Software Application Audit

Design and Code Review Checklist

Javad K. Heshmati <javad@khakbaz.com>

Software Application Audit: Design and Code Review Checklist

by Javad K. Heshmati

Publication date 2006-08-23

Copyright © 2006, 2007, 2008, 2009 Javad K. Hesmati [<http://javad.khakbaz.com/>]

Permission is granted to copy, distribute and/or modify of this document is under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation. A copy of the license is available at <http://www.gnu.org/copyleft/fdl.html>

Table of Contents

Preface	v
1. Introduction	1
1.1. Document Scope	1
1.2. Review Team Participants	1
2. Design Review Checklist	2
2.1. Presentation and coverage aspects	2
2.2. Architecture design	3
2.3. Data design	4
2.4. Program design	7
2.5. Evaluation of program specifications	8
2.6. Process design	9
2.7. Interface with external systems	10
2.8. User interface design	11
2.9. General quality characteristics of design	12
3. Code Review Checklist	15
Glossary	23

Preface

Before programs may be placed in the production system, the source code is reviewed for deficiencies in the areas of validity, security, reliability, performance and operations.

This is an early draft of the guidelines; it is being distributed in the hopes of providing a more transparent and predictable code review process. We do not mean to imply that the things listed here are the only issues we will raise in a review. We will attempt to keep this document up to date, so that over time, it becomes a more useful guide.

Chapter 1. Introduction

1.1. Document Scope

This document is dual purposed; first it is a guideline and checklist for the architecture team performing the architecture and design review; second, it is an attempt to provide development teams with information about what to look for in a review.

1.2. Review Team Participants

The code review participants should remain constant from review to review. Participation by representatives from many groups is required for a successful code review. Those groups include, but are not limited to, project managers, architects and developers. The code must be presented by a developer or group member who is qualified and able to answer technical questions about the code and design. There must be a scribe and a coordinator appointed. At the end of a review, all present must agree on an appraisal of the code. In the event of irreconcilable disagreements, the least positive appraisal that everyone can agree to will be the appraisal. At reviews beyond the first, copies of the diffs and the minutes of all previous meetings should be available, along with a full copy of the current code.

Chapter 2. Design Review Checklist

This chapter addresses general checkpoints for all the tasks associated with the design phase. Where available, specific checklists are also provided separately for different type of applications.

2.1. Presentation and coverage aspects

1. Are all mandatory sections present in the Design Document? If not, is there a reasonable explanation for their absence?
 - Yes
 - No
 - *Remarks:*
2. Have all the required I/O formats (Diagrams, Graphs, Forms, Reports, Screens etc.) been enclosed?
 - Yes
 - No
 - *Remarks:*
3. Have all mandatory models (Process and Data Models) been included?
 - Yes
 - No
 - *Remarks:*
4. Have symbols and notations been used consistently?
 - Yes
 - No
 - *Remarks:*
5. Have all the required I/O formats (Forms, reports, Screens etc.) been enclosed?
 - Yes
 - No
 - *Remarks:*
6. Has the Requirements trace ability matrix been included?
 - Yes
 - No
 - *Remarks:*
7. Have all design assumptions and dependencies been explicitly stated? Are these realistic and reasonable?
 - Yes
 - No

- *Remarks:*

8. Does the design depend upon any assumptions made about the system, which have not been explicitly stated? Are such assumptions acceptable?

- Yes

- No

- *Remarks:*

9. Has the potential reuse components been identified? Else the components to be procured listed?

- Yes

- No

- *Remarks:*

2.2. Architecture design

1. Are the conventions followed consistent with the methodology adopted?

- Yes

- No

- *Remarks:*

2. Is there complete information about the target system environment?

- Yes

- No

- *Remarks:*

3. Is the development environment fully compatible with the target environment? If not, is there a viable plan for porting development outputs to the target system?

- Yes

- No

- *Remarks:*

4. Have all design constraints been identified? Are they acceptable?

- Yes

- No

- *Remarks:*

5. Are design constraints consistent with what is stated in the FSD?

- Yes

- No

- *Remarks:*

6. Does the process model represent a good overview of the proposed system?
 - Yes
 - No
 - *Remarks:*
7. Has an effort been made to include existing objects (developed for other projects)?
 - Yes
 - No
 - *Remarks:*

2.3. Data design

1. Has every logical data structures been converted into appropriate physical structures in the Design Document?
 - Yes
 - No
 - *Remarks:*
2. Has the path for storage of each file/table been specified clearly?
 - Yes
 - No
 - *Remarks:*
3. Have databases and directories been named as per naming conventions?
 - Yes
 - No
 - *Remarks:*
4. Is there complete information about table/record layouts in terms of column/field names, referential integrity details, edit/validation conditions, keys/indexes etc?
 - Yes
 - No
 - *Remarks:*
5. Has a structured process been followed in identifying Primary and Foreign keys?
 - Yes
 - No
 - *Remarks:*
6. Are secondary keys used to the minimum possible extent?

- Yes
 - No
 - *Remarks:*
7. Are the indexes used in the tables/files defined in the Design Document?
- Yes
 - No
 - *Remarks:*
8. Does the design facilitate modification of table/record layouts?
- Yes
 - No
 - *Remarks:*
9. Is there complete information about database/table associated items like views, triggers, stored procedures, Data definition language procedures etc.?
- Yes
 - No
 - *Remarks:*
10. Has a structured process been followed in defining views and triggers?
- Yes
 - No
 - *Remarks:*
11. Has an attempt been made to reduce the number of views and triggers?
- Yes
 - No
 - *Remarks:*
12. Does the design facilitate easy modification or re-definition of views and triggers?
- Yes
 - No
 - *Remarks:*
13. Is there complete information about administrative details like security and access rights, administrative procedures, record locking etc.?
- Yes
 - No
 - *Remarks:*

14. Are the data elements in a file closely related to each other and relevant to the file they are contained in?
 - Yes
 - No
 - *Remarks:*
15. Have efforts been taken to minimize excessive duplication of variables and data-structures?
 - Yes
 - No
 - *Remarks:*
16. Are all data structures initialized properly?
 - Yes
 - No
 - *Remarks:*
17. Have storage requirements been estimated (for whatever period has been stated in the FSD)? Do they take into account peak loads?
 - Yes
 - No
 - *Remarks:*
18. Have all data structures and the operations to be performed on each been identified?
 - Yes
 - No
 - *Remarks:*
19. If databases are envisaged, have they been normalized?
 - Yes
 - No
 - *Remarks:*
20. Are the data structures consistent with the Data Model diagram?
 - Yes
 - No
 - *Remarks:*
21. Can the data structures be linked with user reports and screens?
 - Yes
 - No
 - *Remarks:*

22. Have the size and composition of data structures been estimated? Have provisions been made to guard against overflow?
 - Yes
 - No
 - *Remarks:*
23. Has a data dictionary been established? Can it be used to define data design? Will it facilitate impact analysis of a change?
 - Yes
 - No
 - *Remarks:*
24. Does the design specify clearly what data has to be backed-up and when? Does it specify procedures for restoration?
 - Yes
 - No
 - *Remarks:*
25. Have storage requirements been estimated (for whatever period has been stated in the FSD)? Do they take into account peak loads?
 - Yes
 - No
 - *Remarks:*

2.4. Program design

1. Is each process identified in the design document represented by one or more programs?
 - Yes
 - No
 - *Remarks:*
2. Is each program identified in the design document linkable with a process?
 - Yes
 - No
 - *Remarks:*
3. Have the programs been named as per naming conventions?
 - Yes
 - No
 - *Remarks:*

4. Is there complete information available about the hierarchical organization of programs?
 - Yes
 - No
 - *Remarks:*
5. Do the function (sub-routines) have a high degree of cohesion individually, and a low degree of coupling among themselves?
 - Yes
 - No
 - *Remarks:*
6. Does each function (sub-routine) have a single entry and single exit point?
 - Yes
 - No
 - *Remarks:*
7. Have all defined functions been called at least once? In turn, have all referenced functions been defined?
 - Yes
 - No
 - *Remarks:*
8. Are the numbers of parameters passed between programs within reasonable limits?
 - Yes
 - No
 - *Remarks:*
9. Is there complete information about every interface with external systems?
 - Yes
 - No
 - *Remarks:*

2.5. Evaluation of program specifications

1. Are program specifications in accordance with corresponding process descriptions in the Design Document?
 - Yes
 - No
 - *Remarks:*
2. Are program specifications complete and unambiguous? Will it be possible to easily convert the specifications into code?

- Yes
 - No
 - *Remarks:*
3. Have structured programming constructs been used throughout?
- Yes
 - No
 - *Remarks:*
4. Does the programming language or tool selected, support the data structures and programming constructs used in the specifications?
- Yes
 - No
 - *Remarks:*
5. Have the specifications been written in a manner independent of any specific language or compiler?
- Yes
 - No
 - *Remarks:*
6. Has compound or inverse logic been avoided?
- Yes
 - No
 - *Remarks:*
7. Has provision been made for error handling?
- Yes
 - No
 - *Remarks:*
8. Have exceptional conditions been handled?
- Yes
 - No
 - *Remarks:*

2.6. Process design

1. Is there a correspondence between FSD requirements and atomic processes identified in the Process descriptions?
- Yes

- No
 - *Remarks:*
2. Is the description of processes complete, consistent and correct?
 - Yes
 - No
 - *Remarks:*
 3. Will it be possible to directly generate program specifications from the description of atomic processes?
 - Yes
 - No
 - *Remarks:*
 4. Are detailed specifications available for interfaces between processes, if any?
 - Yes
 - No
 - *Remarks:*
 5. Have all batch tasks been identified? Is the periodicity and dependencies between batch tasks clearly established?
 - Yes
 - No
 - *Remarks:*

2.7. Interface with external systems

1. Is there complete information available about interfaces with different external systems? (It is mandatory to know at least the following: identity of the system, type of interface, contents, media, formats, timing, protocols etc.)
 - Yes
 - No
 - *Remarks:*
2. Have the dependencies/connections/communications between the systems components outlined in the FSD been translated into appropriate interfaces?
 - Yes
 - No
 - *Remarks:*
3. Has an attempt been made to simplify all interfaces?
 - Yes

- No
- *Remarks:*

2.8. User interface design

1. Have screens and reports given by users been retained without change in the design? If not, is there a good explanation for the change?
 - Yes
 - No
 - *Remarks:*
2. Is a separate interface required for the System Administrator / Manager? If so, has this been properly defined?
 - Yes
 - No
 - *Remarks:*
3. Are the screens as per Company/customer standards?
 - Yes
 - No
 - *Remarks:*
4. Is there standardization in the following?
 - Appearance / structuring of screens?
 - Usage of function keys?
 - Availability and appearance of icons (in case of a GUI based application)?
5. Is there complete information available about the hierarchical organization of screens?
 - Yes
 - No
 - *Remarks:*
6. Does the design facilitate easy addition/deletion/modification of screens, in terms of the following?
 - Screen layout?
 - Screen layout?
 - Menus available?
 - Field layout, etc?
7. Is there complete information available about the action associated with each option available on different screens / menus?

- Yes
 - No
 - *Remarks:*
8. Is there complete information available about fields on various screens? (It is mandatory to know the type, width, source, editing associated with each field)
- Yes
 - No
 - *Remarks:*
9. Is there a direct association between screens and processes identified in the process model?
- Yes
 - No
 - *Remarks:*
10. Is there standardization in the appearance / structuring of reports?
- Yes
 - No
 - *Remarks:*
11. Is there complete information available about reports? (It is mandatory to know details such as Layout, content, periodicity, distribution, presentation details etc.)
- Yes
 - No
 - *Remarks:*
12. Does the design facilitate easy addition/deletion/modification of reports?
- Yes
 - No
 - *Remarks:*

2.9. General quality characteristics of design

1. Does the design address all the functional requirements stated in the FSD?
- Yes
 - No
 - *Remarks:*
2. Is the design capable of achieving the performance requirements?
- Yes

- No
 - *Remarks:*
3. Does the design provide for and adequately address all the desirable characteristics stated in the FSD?
- Yes
 - No
 - *Remarks:*
4. Have key assumptions and dependencies been validated (using prototypes, simulation, analysis or any other rational method)?
- Yes
 - No
 - *Remarks:*
5. Have risks arising out of failure of assumptions and dependencies been identified? Has the impact of risks been adequately analyzed?
- Yes
 - No
 - *Remarks:*
6. Have deviations from Company Standards been made? Are these justified and reasonable?
- Yes
 - No
 - *Remarks:*
7. Will it be possible to define details of the Acceptance, System and Integration tests on the basis of the information available in the design?
- Yes
 - No
 - *Remarks:*
8. In the case of Object-oriented projects, does the data design facilitate information hiding?
- Yes
 - No
 - *Remarks:*
9. Has an effort been made to identify different types of errors, failures and exceptions? Have adequate remedial actions been proposed?
- Yes
 - No
 - *Remarks:*

10. Can the final system be tested and supported with existing standard facilities (hardware, software, tools etc.)?
- Yes
 - No
 - *Remarks:*
11. Is the definition of system controls and security adequate?
- Yes
 - No
 - *Remarks:*
12. Are there any TBDs (To Be Determined)? If so, is there a schedule for their resolution?
- Yes
 - No
 - *Remarks:*
13. Have the TBDs identified in earlier documents (e.g.: FSD) been resolved? If not, is there no effect expected on the Design Document?
- Yes
 - No
 - *Remarks:*

Chapter 3. Code Review Checklist

1. Does the code implement its specifications?
 - Yes
 - No
 - *Remarks:*
2. Has the program been divided or grouped into logical segments?
 - Yes
 - No
 - *Remarks:*
3. Is the size of each segment optimal (not too large, not too small)?
 - Yes
 - No
 - *Remarks:*
4. Does each segment have well defined functions?
 - Yes
 - No
 - *Remarks:*
5. On a 10 point scale (10 = Excellent, 1 = Terrible), how would you rate the code on the following parameters:
 - Reliability?
 - Efficiency?
 - Flexibility?
 - Maintainability?
 - Portability?
 - Re-usability?
 - Interpretability?
 - Readability?
6. Does the layout bring out the logical structure of the program?
 - Yes
 - No
 - *Remarks:*
7. By and large, is there only one statement per line?

- Yes
 - No
 - *Remarks:*
8. Have expressions been coded in a readable manner?
- Yes
 - No
 - *Remarks:*
9. Are groups of assignment statements aligned together (around the '=' sign)?
- Yes
 - No
 - *Remarks:*
10. Is there sufficient introductory information about the program in terms of
- Functionality?
 - Files used?
 - Development history?
 - Reason for changes?
11. Has a uniform naming convention been adhered to?
- Yes
 - No
 - *Remarks:*
12. If so, which of the following standards or conventions has been adhered to?
- Company Standard?
 - Client specific?
 - Project specific?
13. Have the following guidelines been adhered to while declaring variables?
- One variable per line (by and large)?
 - Variables of the same type declared together?
 - Variables declared in alphabetical order/logical order/order of data types?
14. Have file declarations been made properly?
- Yes
 - No
 - *Remarks:*

15. Have comments been used to describe peculiarities in any complex data structure?

- Yes
- No
- *Remarks:*

16. Are there any un-initialized local variables?

- Yes
- No
- *Remarks:*

17. Are global variable definitions consistent across modules?

- Yes
- No
- *Remarks:*

18. Has an effort been made to minimize global variables and constants?

- Yes
- No
- *Remarks:*

19. Are global variables being used appropriately?

- Yes
- No
- *Remarks:*

20. Is there total agreement between the parameters (passed from the calling module) and arguments (used in the called module) in terms of

- Number?
- Type?
- Order/sequence?

21. Does the program explicitly declare all the header files used?

- Yes
- No
- *Remarks:*

22. Have structured programming constructs been adopted?

- Yes
- No
- *Remarks:*

23. Have only standard features of the language/tool been used? If not, has this been justified and highlighted through appropriate comments?
- Yes
 - No
 - *Remarks:*
24. Have non-standard practices been resorted to? If so, have they been justified?
- Yes
 - No
 - *Remarks:*
25. Has excessive nesting of loops and conditions been avoided?
- Yes
 - No
 - *Remarks:*
26. Have all unnecessary branching of loops been avoided?
- Yes
 - No
 - *Remarks:*
27. Have negative and complicated conditions been avoided in IF statements and loops?
- Yes
 - No
 - *Remarks:*
28. Has the use of GOTO statement been avoided? If not, is it as per the practice advocated?
- Yes
 - No
 - *Remarks:*
29. Has the use of backward jumps and jumps out of loops been avoided?
- Yes
 - No
 - *Remarks:*
30. Are requests for interactive inputs accompanied by appropriate and helpful prompts?
- Yes
 - No
 - *Remarks:*

31. Are Help and Abandon provisions available throughout the program in the same consistent manner?

- Yes
- No
- *Remarks:*

32. Are the input formats uniform and simple (easy to comprehend and use)?

- Yes
- No
- *Remarks:*

33. Is input being validated for valid as well as invalid values?

- Yes
- No
- *Remarks:*

34. Is the user being cautioned about the possible consequences of his/her actions?

- Yes
- No
- *Remarks:*

35. Are outputs being checked for validity before display or printing?

- Yes
- No
- *Remarks:*

36. Does the program provide preview facilities before printing?

- Yes
- No
- *Remarks:*

37. Are all implementation aspects hidden from the user?

- Yes
- No
- *Remarks:*

38. Does the program prevent unauthorized usage by checking for passwords and IDs?

- Yes
- No
- *Remarks:*

39. Does the program check for security permissions before accessing key resources?

- Yes
- No
- *Remarks:*

40. Does the program handle issues related to unauthorized usage in the proper manner?

- Yes
- No
- *Remarks:*

41. Are files and records locked before updating to protect data integrity and consistency?

- Yes
- No
- *Remarks:*

42. Has *hard-coding* of the following been avoided?

- Device parameters?
- Constants and literals?
- Error messages?
- Help messages?
- Data/file dump paths?

43. Has repeating code been replaced by a function?

- Yes
- No
- *Remarks:*

44. Has CASE construct been used for multi-way selection?

- Yes
- No
- *Remarks:*

45. Are messages appropriate to the context?

- Yes
- No
- *Remarks:*

46. Does the program wait for the user to acknowledge the message before clearing it?

- Yes
- No

- *Remarks:*

47. Do messages provide a reference to other sources through appropriate mechanisms?

- Yes
- No

- *Remarks:*

48. Do error messages appear at a constant location on the screen and are consistent in presentation aspects?

- Yes
- No

- *Remarks:*

49. Do error messages convey what is wrong in a simple and understandable manner?

- Yes
- No

- *Remarks:*

50. Do error messages provide enough information to resolve the error made?

- Yes
- No

- *Remarks:*

51. Is HELP available through the same key or icon throughout the application?

- Yes
- No

- *Remarks:*

52. Are all source code lines reachable? (i.e., there is no *dead code*)

- Yes
- No

- *Remarks:*

53. Is there a conscious attempt to minimize the I-O requests?

- Yes
- No

- *Remarks:*

54. Are data types being mixed?

- Yes
- No

- *Remarks:*

55. Has an effort been made to use efficiency features provided by the compiler or tool used?

- Yes
- No

- *Remarks:*

56. Can the comments in the source code be described as follows?

- Appropriately placed?
- Separated from logical and physical flow of program?
- Unambiguous?
- Precise yet descriptive and adequate?
- Accurate and apt?

57. Do the comments describe what is being done in the code, instead of how?

- Yes
- No

- *Remarks:*

58. Do the comments add to the understandability of the code in a significant way?

- Yes
- No

- *Remarks:*

59. Will it be possible to change the comments in an easy manner?

- Yes
- No

- *Remarks:*

60. Is the code free from possibility of covert channels (unintentional additional code) and Trojan code?

- Yes
- No

- *Remarks:*

Glossary

Terms

FSD

Functional Specification Document