Software Engineering 2

Inspection Document

Authors
Paolo Paterna, Lara Premi

Politecnico di Milano, Italy
Email: paolo.paterna@mail.polimi.it, lara.premi@mail.polimi.it

5th Jan 2016

Contents

1 Classes Analyzed

1.1 Class Name and Location

The only class assigned to our group is `Registered Resources` located in `appserver/transaction/jts/src/main/java/com/sun/jts/CosTransactions` package.

In particular, we have assigned the `distributeCommit()` method.

1.2 Class Functional Role

Thanks to the Javadoc and the code, we have understood that the class `Registered Resources` , in which the method assigned to us is located, works with some entities, called "Resources", and it generates different types of message, that refer to those entities.

2 Issues Checklist

2.1 Naming Conventions

1. Class name is pretty self-explanatory. It handles all the registered resources.

   Reading the Javadoc inside the class file adds more information: this class handles some Resource Objects that are involved in a transaction. These objects are stored in a list together with their state relative to such transaction.

   Since an instance of this transaction can be accessed by multiple threads, the Javadoc suggests to use serialization in the implementation.

   All the information relative to an instance of this class should be reconstructable in case of system failure, so they are recorded.

```
/**
* The RegisteredResources class provides
   operations that manage a list
* of Resource objects involved in a
   transaction, and their states relative
* to the transaction. Resource references are
   stored in lists as there is no
* way to perform Resource reference
   comparisons. As an instance of this
* class may be accessed from multiple threads
   within a process,
* serialisation for thread-safety is necessary
   in the implementation.
* The information recorded in an instance of
   this class needs to be
* reconstructible in the case of a system
   failure.
*
* @version 0.02
*
* @author Simon Holdsworth, IBM Corporation
*
* @see
*/
```

- Line 112: a Resource called laoResource;
- line 774: a boolean called rmErr, maybe removedError;
- line 901: an integer called i that corresponds to the number of the current resource; it used to a loop but maybe it should be used a better name, since it's not only a loop variable;
- line 1470: this method is the distribution of one phase commit, not the commit itself.

Exception called with general name.

- Line 262: e;
- line 304: exc;
- line 488: exc;
- line 538: exc;
- line 563: ex2;
- line 774: exc;
- line 862: e with also empty statement;
- line 1049: exc;
- line 1124: exc;
- line 1259: exc;
- line 1426: exc;
- line 1577: exc;
- line 1601: e;
- line 1675: e.

2. According to the function that we have to inspect, there are no one-character variables used.

3. There is only one class called "RegisteredResources", that is a noun with capitalized letters for every word.

4. Although our function belongs to a class, there are no interface to consider.

5. Methods are all verbs and all of them are written in camelcase.

6. The variable "private static boolean lastXAResCommit" is a bit ambiguous: we don't know if the two letters "XA" define an abbreviation or something else. In the other cases, all the attributes are written in the right way.

7. Constants LOG_SECTION_NAME and HEURISTIC_LOG_SECTION_NAME are all in uppercase.

2.2 Indention

8. Although the method code has a clear order, there are some lines in which the tab is used, some lines in which both tabs and spaces are used and some lines in which there are used more or less than four spaces. According to the first case:

   - lines from 758 (included) to 764 (included);
   - lines from 766 (included) to 772 (included);
   - lines 822 and 823;
   - lines from 829 (included) to 831 (included);
   - lines 841 and 842.

   According to the second case:

   - line 828;
   - line 832;
   - lines from 847 (included) to 850 (included).

   According to the third and final case:

   - line 766;
   - lines from 790 (included) to 804 (included);
   - line 807;
   - lines from 809 (included) to 811 (included);
   - lines from 815 (included) to 822 (included);
   - line 824;
   - line 827;
   - line 833;
   - lines from 835 (included) to 838 (included);

- line 840;
- line 843;
- lines 845 and 846;
- lines from 854 (included) to 856 (included);
- line 860;
- line 862;
- lines from 866 (included) to 869 (included);
- lines from 871 (included) to 875 (included);
- line 892;
- lines from 898 (included) to 903 (included);
- line 905;
- lines 917 and 918;
- lines from 921 (included) to 924 (included).

9. Our class contains 818 tabs that are spread every now and then.
   In particular, in the assigned method, there are tabs in:

   - from line 758 included to line 764 included;
   - from line 766 included to line 772 included;
   - from line 822 included to line 825 included;
   - from line 828 included to line 832 included;
   - from line 841 included to line 844 included;
   - from line 847 included to line 850 included;
   - line 868;
   - line 869;
   - line 873;
   - line 874;
   - line 875.

2.3 Braces

10. Only in the "if" branch of the first block "try-catch", the Allman style is used. In the other cases, it's used the Kernighan and Ritchie style.

11. Our class has 16 "if" condition with only one statement not enclosed in a curly braces block `https://regex101.com/r/cQ9lH5/1` .

    - Line 219;
    - line 602;
    - line 623;
    - line 714;
    - line 890;
    - line 928;
    - line 930 (also else misses curly braces);
    - line 988;
    - line 1151;
    - line 1179;
    - line 1184 (also else misses curly braces);
    - line 1186;
    - line 1506;
    - line 1610 (also else misses curly braces);
    - line 1614 (also else misses curly braces);
    - line 1711.

2.4 File Organzation

12. Blank lines and optimal comments are used to separate sections.
13. There are lines that exceed 80 characters:

    - line 114;
    - line 126;
    - line 133;
    - line 169;
    - line 236;
    - line 244;
    - line 249;
    - line 297;

- line 310;
- line 311;
- line 329;
- line 603;
- line 693;
- line 741;
- line 777;
- line 779;
- line 780;
- line 824;
- line 829;
- line 830;
- line 843;
- line 848;
- line 849;
- line 869;
- line 875;
- line 892;
- line 893;
- line 913;
- line 924;
- line 1016;
- line 1080;
- line 1082;
- line 1083;
- line 1107;
- line 1108;
- line 1131;
- line 1135;
- line 1153;
- line 1155;

- line 1181;
- line 1218;
- line 1246;
- line 1294;
- line 1499;
- line 1562;
- line 1571;
- line 1602;
- line 1610;
- line 1614;
- line 1641;
- line 1645;
- line 1646;
- line 1656;
- line 1660;
- line 1661;
- line 1694;
- line 1732.

14. When the line length exceed 80 characters, it is not longer than 120 characters.

2.5 Wrapping Lines

15. No break lines are performed if there isn't an operator or a comma.

16. Not always the higher-level breaks are used. In fact, it happens according to the lines listed below:

    - lines 730 and 731;
    - lines from 760 (included) to 763 (included);
    - lines from 768 (included) to 771 (included);
    - lines 779 and 780;
    - lines from 822 (included) to 824 (included);
    - lines from 841 (included) to 843 (included);

- lines from 847 (included) to 850 (included);
- lines 868 and 869;
- lines from 873 (included) to 875 (included).

17. There are a lot of unaligned lines with wrong tabulation:
    - line 239;
    - line 242;
    - line 243 (with whole "if" block);
    - line 300 (maybe copied from Line 239);
    - line 303 (maybe copied from Line 242);
    - line 304 (with whole "if" block, maybe copied from Line 243);
    - line 367;
    - line 523;
    - line 533;
    - line 638 (whole "if" block);
    - line 646 (whole "if" block);
    - line 826 (one unneeded space);
    - lines 845-847 (one unneeded space);
    - line 868;
    - lines 871-873;
    - lines 917-924;
    - lines 1106-1111;
    - line 1130;
    - lines 1293-1298;
    - line 1606 (whole "if" block);
    - lines 1693-1698.

2.6 Comments

18. Maybe there could be more comments referring to the sections about the logs, but in the other cases the comments are used to adequately explain what the class, method and block of code are doing.

19. There are lots of commented out code, often with little or no comments.

- Lines 113-118 comment: IASRI 4662745;
- lines 123-126 comment: moved in another class: Configuration.java;
- lines 161-170 no comment;
- line 274 comment: code not reachable;
- line 335 comment: code not executed;
- line 369 no comment;
- lines 387-395 has javadoc but no comment about why is commented out;
- lines 498-499 no comment;
- line 644 no comment;
- lines 704-712 no comment;
- lines 827-833 no comment;
- lines 877-896 no comments;
- lines 1449-1459 comment: Admin package is not supported anymore;
- lines 1487-1504 no comment;
- lines 1595-1600 comment: IASRI 4722883;
- lines 1745-1754 comment IASRI 4662745.

2.7 Java Source Files

20. Referring to the file containing the class in which there is our function, this constraint is not violated.

21. No public class constructor is specified in the file. There are only two package friendly constructors available at lines 152 and 183 after class variables.

22. In some cases the Javadoc is not completed:

    - in the first constructor, the Javadoc contains only one of the two parameters;
    - in the second constructor, the Javadoc doesn't contain the specified parameters;
    - there is no Javadoc for the method "getLAOResource";

- referring to our function, the third class of exception is not contained in the Javadoc;
- in the function called "distributedForget", two parameters are not specified in the Javadoc;
- in the function called "commitOnePhase", two classes of exception are not specified in the Javadoc.

23. Where the Javadoc is present, it's almost complete. Some @param are missing in one constructor.

2.8 Package and Import Statements

24. According to our function, the package statement is the first non-commented statement. Then it is followed by the import statements.

2.9 Class and Interface Declaration

25.
- Class documentation comment: OK.
- Class statement: OK.
- Class implementation comment: optional, not present in this class.
- Class variables:
  - no public vars OK;
  - no protected vars OK;
  - no package vars OK;
  - some private vars mixed with static vars KO ;
- Instance variables:
  - no public vars OK;
  - no protected vars OK;
  - no package vars OK;
  - some private vars mixed with static vars KO;
- Constructors: OK.
- Methods: OK.

26. Only the method "getLAOResource" is not near the functions that work with the resources. The other methods are grouped by the functionality: at first, there are the two constructors; then the methods referring to work with the resources and finally there are the methods regarding the distribution of the messages.

27. Exceptions are often duplicated. Sometimes it is obvious that there was a copy-paste work.

2.10 Initialization and Declaration

28. All the attributes of the class are private. The static variable Logger, called "_logger", is not private, but it should be in this way by default.

    The type of the two arraylists that are attributes of the class is not specified: it should be written to be clear.

    Referring only to our class, the visibility of its private method is correct because it is used only by the class itself.

29. All variables in the method "distributeCommit" are declared in the proper scope.

30. In this class, the constructors are not used.

31. All object references in the method "distributeCommit" are initialized before their usage.

32. Only in the method called "commitOnePhase" there are two variables that are declared but not initialized. These variables are "retry_limit" and "no_of_attempts".

33. Some variables in the method "distributeCommit" are not declared at the start of a block:

    - line 721;
    - line 722;
    - line 754;
    - line 755.

2.11 Method Calls

34. In this class, only a method is used, the private one called "distribute-Forget". In this case, the parameters are presented in the correct order.

35. All methods have a unique name.

36. In this class, only a method is used, the private one called "distribute-Forget". In this case, the method returned values are used properly.

2.12 Arrays

37. Referring to the class of our method, there are no arrays here.

38. Referring to the class of our method, knowing that there are no arrays in this class, other collections indexes are prevented from going out-of-bounds.

39. Referring to the class of our method, there are no arrays here.

2.13 Object Comparison

40. Referring to the class of our method, there are some lines in which is not used "equal", but the comparison of two objects is done through "==":

    - line 463;
    - line 470;
    - line 599;
    - line 608;
    - line 629;
    - line 635;
    - lines 730 and 731;
    - lines 1235 and 1236;
    - lines 1531 and 1532;
    - line 1610;
    - line 1614.

2.14 Output Format

41. The only output in our method is in the logs. No spelling or grammatical errors are present, only some abbreviations.

42. According to the class of our method, the error messages are comprehensive, thanks also to the comments written that are referred to the code. But maybe it should be more meaningful to add others comments to explain better what it happens and how to solve the problems.

43. Some spaces in logs are replaced with underscores.

2.15 Computation Comparison and Assignments

44. Only in the method "involved()", that return a boolean value, it's used the Brutish Programming: we are referring to the lines 410, 411 and 412. Maybe, referring to two "catch" blocks (the first "catch" located from line 1049, included, to line 1140, included; the second "catch" located from line 1577, included, to line 1700, included), it could be used a switch, instead of "if", "else if" and "else" conditions, to simplify the reading by other programmers.

45. The parenthesizing and the operators order are good; there aren't numerical operations.

46. Referring to the class of our method, there are some lines in which the parenthesis are not used to avoid operator precedence problems:

    - line 252;
    - line 305;
    - line 312;
    - line 463;
    - line 470;
    - line 551;
    - line 568;
    - line 635;
    - line 776;
    - lines 779 and 780;
    - lines 790, 791 and 792;
    - lines 806 and 807;
    - line 852;
    - lines 1067, 1068 and 1069;
    - lines 1082 and 1083;
    - lines 1087 and 1088;
    - line 1113;
    - lines 1261 and 1262;
    - line 1277;
    - line 1602;

- lines 1622 and 1623;
- line 1665.

47. In the class, no divisions are present.

48. Integer arithmetic is used appropriately. Maybe, at lines 918 and 1181, referring to the two additions, overflows could happen.

49. Comparison between boolean is made correctly.

50. Referring to lines 243, 304, 488, 538, 563, 653, 774, 1049, 1259, 1365 and 1577, it's better to be as specific as possible. So it's not correct to catch the Throwable class: it's too much general. Maybe it's better to use the Exception class, that it's general too: according to the Java documentation, "An Error is a subclass of Throwable that indicates serious problems that a reasonable application should not try to catch".

    At lines 262, 322, 862, 1124, 1287, 1426 and 1675, it's not specified how to manage the exception.

51. No implicit conversions are made in the class.

2.16 Exceptions

52. All the relevant exceptions are caught: in fact, in the catch blocks, it is specified a superclass of exceptions, that is the Throwable class; in this way, it's very difficult not to cover all the exceptions that can happen in the class.

53. Every catch block handles correctly the thrown exception.

2.17 Flow of Control

54. There are no "switch" statements in the class of our method.

55. No "switch" statements are present in the class.

56. Referring to the class of our method, all the loops are correctly formed, with the appropriate initialization, increment and termination expressions.

2.18 Files

57. In the class of our method, there are no files used.

58. In the class of our method, there are no files used.

59. In the class of our method, there are no files used.

60. In the class of our method, there are no files used.

2.19 Other Problems Encountered

The class file is very long: in fact, it has 1758 lines. Probably dividing it in more than one class would improve readability and maintainability.

Some "catch" statements contain multiple `else if` statement; in this case using a "switch" can greatly improve readability and the catch block can be shortened by adding more method to handle this massive use of else if statement that make the method much longer.

2.20 Hours of Work

In order to write this document, we have done the following hours of work:

- Paolo Paterna: 20 Hrs;
- Lara Premi: 20 Hrs.