

Version: 1.0.0

Release date: 28-12-2016

Riccardo Cattaneo 873647

Fabio Chiusano 874294

Code Inspection

Table of Contents

2 Classes assigned 3

3 Functional role of assigned set of classes 4

4 Issues found by applying the checklist 6

5 Other problems 7

# Classes assigned

We have been assigned the following classes:

* ../apache-ofbiz-16.11.01/applications/party/src/main/java/org/apache/ofbiz/party/content/PartyContentWrapper.java

# Functional role of assigned set of classes

OFBiz is an Enterprise Resource Planning (ERP) System written in Java and houses a large set of libraries, entities, services and features to run all aspects of a business.

The OFBiz documentation can be found at the following link:

<http://ofbiz.apache.org/documentation.html>

From the name of the class we have to analyse, that is PartyContentWrapper, we can deduce that is a wrapper for the content of a Party. The first question we need to answer is: what is a Party in OFBiz? From the documentation, specifically from <https://ofbiz.apache.org/apache-ofbiz-project-overview.html>, we’ve learned that:

A Party can be either a Person, or a group of Parties. A Party Group could be a company, an organization within the company, a supplier, a customer, and so forth. Information that describes Parties or is directly related to Parties is contained in these entities.

One type of related data is Contact Mechanisms such as postal addresses, phone numbers, email addresses, internet URLs. Another is Roles that the Party acts in such as Customer, Supplier, Employee, Manager, Merchandiser, etc. Generally a single party will interact with different parts of the system in many different roles.

Another type of data that fits into the Party category is information about communication and agreements between Parties. This gets into the area of relationship management and also includes information about issues or trouble tickets that a Party may have. These entities are used along with the Work Effort entities to plan and track the research and resolution of such issues.

We can summarize by saying that a Party:

* can be either a Person or a group of Parties;
* contains the following data:
  + Contact Mechanisms such as postal addresses, phone numbers, email addresses, internet URLs;
  + Roles that the Party acts, such as Customer, Supplier, Employee, Manager;
  + Information about communication and agreements between Parties; these entities are used along with the Work Effort entities to plan and track the research and resolution of issues such as trouble tickets.

There isn’t a single javadoc comment in the whole file and there are few comments, therefore it won’t be probably easy to understand what the methods do.

However, the comment before the class declaration is:

/\*\*

\* WorkEffortContentWrapper; gets work effort content for display

\*/

This is confusing, since there is no WorkEffortContentWrapper in our PartyContentWrapper class. However, by checking the WorkEffortContentWrapper.java file, we found that the same comment is present in that class too, along with a lot of copied code. Therefore we can conclude that our class was not written from zero, but starting with a copy and paste of the WorkEffortContent class and then replacing each occurrence of *WorkEffort* with *Party*. This suggests that the functionalities may be the same.

In the WorkEffortContentWrapper.java file there are more comments than in its copied class, therefore it can be useful to see how many methods they have in common, so that we can read the javadoc comments in WorkEffortContentWrapper in order to understand them. It turns out that **all** methods in PartyContentWrapper.java have a corresponding method in WorkEffortContentWrapper.java that shares a huge part of code. This is a sign of bad code reuse and software design. However, these problems should have been addressed previously in the development and now we are interested in code issues only. Moreover, the ContentWrapper interface, that is the interface implemented by both PartyContentWrapper and WorkEffortContentWrapper, has only one of those methods.

Analysing the methods of the class (and therefore its functionalities), we understand that PartContentWrapper is a wrapper for a Party, that has convenient methods to get its content and display it.

# Issues found by applying the checklist

We should check:

* Naming conventions;
* Indention;
* Braces;
* File organization;
* Wrapping lines;
* Comments;
* Java source files;
* Package and import statements;
* Class and interface declarations;
* Initialization and declarations;
* Method calls;
* Arrays;
* Object comparison;
* Output format;
* Computations, comparisons and assignments;
* Exceptions;
* Flow of control;
* Files.

Some issues:

* Almost total absence of javadoc comments;
* This class was at first entirely copied from WorkEffortContentWrapper, as suggested from the comment before the class declaration that went unchanged in the copy and paste process. Copying code may be faster but may actually result in some bugs going unnoticed because, generally, modifying involves less concentration that writing from zero. Some issues found in this class are probably present in WorkEffortContentWrapper too.

# Other problems