International Journal of Multidisciplinary Research and Publications
ISSN (Online): 2581-6187

Data Exchange Using JavaScript Object Notation and REST API

Ester Lumba[1]
[1]Department of Informatics, Bunda Mulia University, Jakarta, DKI Jakarta, Indonesia
Email address: l0178@lecturer.ubm.ac.id

Abstract**—** *Currently, computer applications can be accessed from various platforms and various devices. One of the very popular platforms is Android. Developing Android-based applications has its own challenges. Besides requiring hardware devices with quite high specifications, they must also learn various programming languages. On the client side the programmer must understand XML for layout and Java for programming logic. On the server side must master PHP and SQL for data processing. To exchange data, the developer must also master the commonly used data formats such as Java Script Object Notation. Android application development for monitoring project progress to exchange data from client to server and vice versa must handle HTTP methods. This study aims to examine the process of exchanging data on Android applications using the Java Script Object Notation and REST API with the Retrofit library. The application development method uses extreme programming.*

Keywords**—** *Android, API, JSON, REST, Retrofit.*

I. INTRODUCTION

The use of mobile devices is increasing day by day. For programmers or companies engaged in Information Technology, of course, they have their own challenges in providing mobile-based applications and solutions. According to research conducted by DataReportal, the number of connected mobile devices in Indonesia in January 2022 was 370.1 million, an increase of 3.6 percent or 13 million from the same period in the previous year [1]. According to Simon Kemp, author of a report on DataReportal that data from GSMA Intelligence shows cellular connections in Indonesia in early 2022 there were 370.1 million. In January 2022, the population in Indonesia now reaches 277.7 million. This means that data from GSMA Intelligence shows that mobile devices in Indonesia are equivalent to 133.3 percent of the total population in January 2022. Of these cellular users, 96% of their activities are to access the Internet.

Developing mobile application applications such as Android has its own challenges. Development requires hardware with high specifications, and must use various programming languages. On the client side, programmers must understand XML for layouts and Java or Kotlin as programming languages. On the server side, you must master PHP and SQL to perform data processing [2].

As a reference in the research, the author conducted a literature review on the journals of several previous related studies. Like the research conducted by Jiri Hradil, Vilém Sklenak with the title “Practical Implementation of 10 Rules for Writing REST APIs”. This research implements 10 REST API rules that are applied to small business applications that can be accessed from all over the world. The API is implemented in JSON format. The main purpose of making the API is to make it easier for application users to issue invoices so they can be sent to customers [3]. Further research was carried out by Jaydevi Bhade and Prof. Himanshu Yadav with the title "Evaluation of Android Networking Libraries". This study analyzes four libraries, namely HTTP Client, Volley, Retrofit and Fast Android Networking. Researchers carried out comprehensively on the four libraries in the form of a matrix [4].

Based on these problems, the researchers conducted research on data exchange in Android applications using Java Script Object Notation and REST API with Retrofit library. The research question is "How to exchange data using JSON and REST API with Retrofit on Android Applications?"

The purpose of this research is to build an Android application to exchange data from client to server using JSON and REST API with Retrofit Library.

JSON stands for JavaScript Object Notation and is used as a means to store and transfer structured data over network media. JSON has a data structure that is simple, easy to read, understand and easy to generate by computer programs. The convenience offered by JSON is the reason why it is often used in API creation. This JSON format is a subset of JavaScript Programming, according to Standard ECMA-262 3rd Edition - December 1999 [5] [6]. The JSON data format is independent of any programming language such as C/C++, Java, Python C# and so on. JSON has two structures, namely a collection of key/value pairs and an ordered list of values as in an array. The object starts with an opening curly brace, namely { and at the end it is closed with a closing curly brace, namely }[5]. The following is an example of a representation of the JSON data format to describe the lecturer object.

```
{
    "nidn": "0307345",
    "name": "Vanda Christie",
    "teach": {
       "code": "IF-005",
       "course": "Mobile Programming",
       "credit": "3"
    },
    "mobile_phone": [
       "0813 2258-3888",
       "0819 8545-4258"
    ]
 }
```


Ester Lumba, “Data Exchange Using JavaScript Object Notation and REST API,” *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 5, Issue 3, pp. 42-47, 2022.

In the JSON data structure, each name is followed by a colon: and each key/value pair is separated by a comma, namely. The general form of the JSON object data structure is shown in Figure 1.

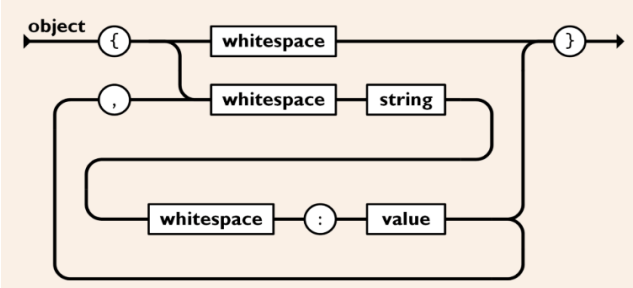

Fig. 1. JSON object structure representation.

Representational State Transfer or commonly abbreviated as REST is a standard of web-based architecture. REST uses the HTTP protocol for data communication.

In the REST architecture, the server side provides resources or data and the client side accesses and displays these resources [7]. Each resource is identified by URIs (Universal Resource Identifiers) or global IDs. JSON or XML is used to represent REST resources [8]. Figure 2 shows how the REST API works in a application.

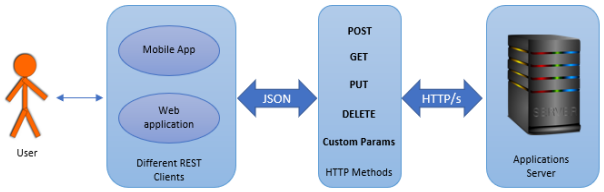

Fig. 2. REST API Architecture

In general, the HTTP method used in the REST API is shown in Figure 4 above. The GET method is usually used to read resources from a REST server. The POST method is usually used to create new resources on a REST server [7]. While the PUT method is generally used to update resources on the REST server. Then the DELETE method, is used to delete the resource from the REST server. The last one is the OPTIONS (CUSTOM PARAMS) method, usually used to get operations supported by a resource on a REST server.

Android is an open-source mobile operating system. This operating system was originally created or developed by Android, Inc [9]. But in 2005, it was acquired and developed by Google. Android OS is used for tablet computers and smartphones. In Palo Alto, Andy Rubin, Rich Miner, Nick Sears, and Chris White founded Android in October 2003. Google acquired Android Inc. on August 17, 2005, however, Rubin, White and Miner remained with the company [10].

Android can be thought of as a software stack. At each layer of the stack, consists of programs that support specific functions of an operating system. The Linux Kernel is the foundation of the Android platform and sits at the bottom of the Android architecture. To build the Android system, Google uses the Linux kernel version 2.6. This kernel includes memory management, power management, security settings and some of the hardware drivers. The next layer is the Hardware Abstraction Layer (HAL). This layer consists of several libraries for implementing hardware component interfaces such as camera modules or Bluetooth. Next up is the Native C/C++ layer and the Android Runtime. Each application on Android will run in its own process and with an instance of the Android Runtime. Android system services and components are built from native libraries written in C and C++. The Java API Framework layer is a basic tool for developers written in the Java language. Developers do not need to know in detail about the API for Android application development purposes. This API is used to build the UI of the application such as buttons, menus and other UI components. The API is also used for resource allocation from smartphones, managing switching between processes and managing data accessibility in other applications. The upper layer consists of two parts, namely System Apps and User Apps. System Apps are the default applications of smartphones such as email, SMS, calendar, Internet browser and contacts applications. While User Apps are applications that are built by Android application developers [11]. Figure 3 shows the architecture of Android.

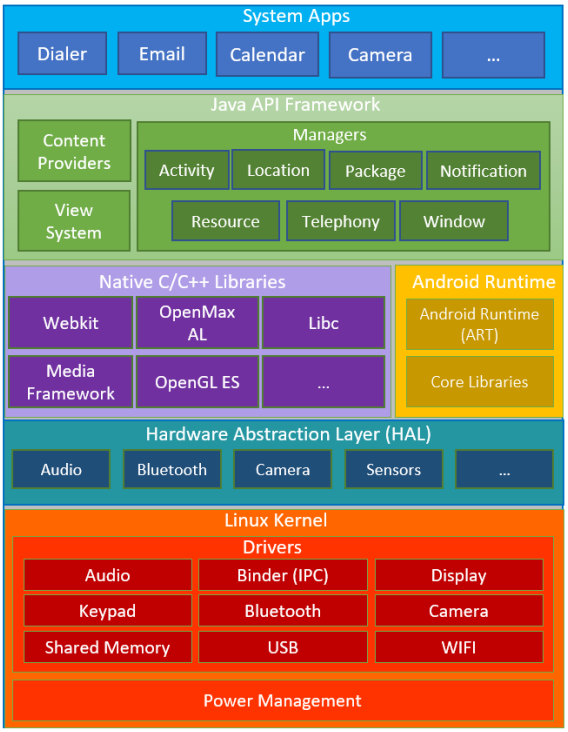

Fig. 3. Android application architecture layers.

II. METHODOLOGY

The research method begins with conducting a literature study using both books and journals related to the research title. The application development method uses Extreme Programming (XP). The case study used is an application for monitoring project work activities by project managers. Applications are made using Android Studio using the Retrofit library. XP is one of the methods in software development.

Ester Lumba, "Data Exchange Using JavaScript Object Notation and REST API," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 5, Issue 3, pp. 42-47, 2022.

The XP method has the characteristics of being fast, efficient, low risk, flexible, predictable and scientific. This model uses an Object-Oriented approach. In the Extreme Programming method, small and medium-sized teams can be formed [12]. Its purpose is to deal with unclear requirements and in the event of very rapid changes in requirements. In the Extreme Programming method, there are 4 phase that must be carried out by developers, as shown in Figure 4 below:

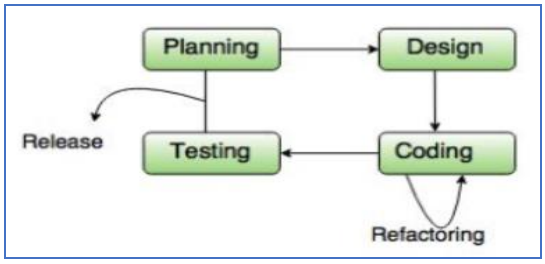

Fig. 4. Extreme programming phase

To perform requirements, analysis and design processes, the technique used for documentation and modelling is the Unified Modelling Language (UML). UML is used to facilitate the presentation of a problem in the form of diagrams and facilitate understanding of the needs and models of the system.

1. Planning Phase

At this phase, the researcher analyzes why a system or application for monitoring the activities of this project needs to be developed. The things that are done are making System Requests, Business Needs, Business Values, non-functional requirements, functional requirements, making use case diagrams and use case descriptions. At this stage the researcher also makes use case diagrams for project activity monitoring applications. User interaction with the application is shown in the Use case diagram as shown in Figure 5 below:

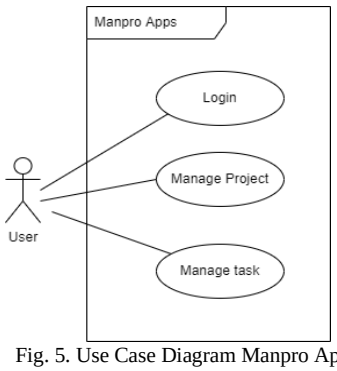

Fig. 5. Use Case Diagram Manpro Apps

Based on the use case diagram shown in Figure 5. above, the next step taken by the researcher is to make a description of each use case. Use case descriptions are made based on the order of the ID of each use case.

2. Design Phase

At this phase the researcher makes activity diagrams and class diagrams as well as a representation of the database used in this study. Figure 6 shows the class diagram used in a project completion monitoring application by a project manager. Consists of three classes, namely the user class which is used to form the user object. Then the Project class is used to create objects from the project and the ProjectList class is used to hold a list of projects that will be monitored by the project manager.

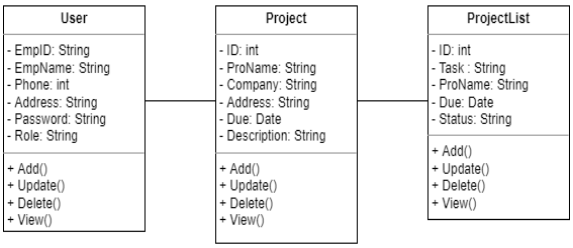

Fig. 6. Class Diagram Manpro Apps

3. Coding Phase

The third phase of Extreme Programming is coding. At this phase the researcher makes program code using Java language for program logic and XML as layout design language. This project monitoring application has a structure as shown in Figure 7 below:

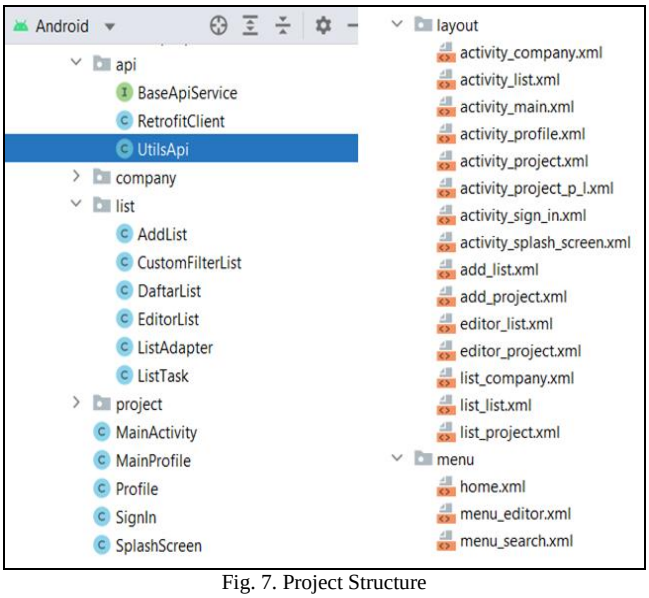

Fig. 7. Project Structure

In the project structure shown in Figure 7 above, it is divided into two, namely the java and res folders. The java folder contains program logic written in Java programming language and res contains layouts written in XML. Of all the application projects made, the researcher will only discuss CRUD operations on the list of projects monitored by the project manager. To facilitate communication between the Android application and the server, the researchers created classes and interfaces using the Retrofit library. The first thing to do is create a BaseApiService interface to bridge the Android application with the database. In the Retrofit library, data exchange is carried out using JSON. This interface defines the POST, GET and form URL endcode methods

Ester Lumba, "Data Exchange Using JavaScript Object Notation and REST API," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 5, Issue 3, pp. 42-47, 2022.

belonging to the http protocol from the Retrofit library. The BaseAPIService interface is shown in Figure 8 below:

```
import okhttp3.ResponseBody;
import retrofit2.Call;
import retrofit2.http.Field;
import retrofit2.http.FormUrlEncoded;
import retrofit2.http.GET;
import retrofit2.http.POST;
import retrofit2.http.Query;

public interface BaseApiService {
    @FormUrlEncoded
    @POST("login.php")
    Call<ResponseBody> loginRequest(@Field("empid") String nik,
                                    @Field("password") String password);

    @GET("get_company.php")
    Call<List<Company>> getCompany(
            @Query("key") String keyword);

    @FormUrlEncoded
    @POST("add_list.php")
    Call<ListTask> insert_list(@Field("project") String project,
                               @Field("task") String task,
                               @Field("due") String due,
                               @Field("status") String status);

    @POST("get_list.php")
    Call<List<ListTask>> getList();

    @FormUrlEncoded
    @POST("update_list.php")
    Call<ListTask> update_list(@Field("key") String key,
                               @Field("id") int id,
                               @Field("project") String project,
                               @Field("task") String task,
                               @Field("due") String due,
                               @Field("status") String status);

    @FormUrlEncoded
    @POST("delete_list.php")
    Call<ListTask> delete_list(@Field("key") String key,
                               @Field("id") int id);
```

Fig. 8. BaseApiService interface

Then create a UtilsApi class that is used to connect web services, namely files written using the PHP programming language. Figure 9 below shows the UtilsApi class script.

```
public class UtilsApi {
    public static final String BASE_URL_API = "http://10.0.2.2/api/";
    public static BaseApiService getAPIService(){
        return RetrofitClient.getClient(BASE_URL_API)
                .create(BaseApiService.class);
    }
}
```

Fig. 9. UtilsApi class

Then the researcher created a RetrofitClient class. In this class, the converter is configured using GSON to serialize the data. By default, Retrofit can only deserialize HTTP bodies into ResponseBody OkHttp type and can only accept RequestBody types for @Body. The RetrofitClient class is shown in figure 10.

Of all the project monitoring applications that were built, the researchers only experimented with exchanging data using the Retrofit library in the creation of tasks. The view for creating a new task is shown in Figure 11.

```
import okhttp3.OkHttpClient;
import okhttp3.logging.HttpLoggingInterceptor;
import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory;

public class RetrofitClient {
    public static Retrofit retrofit = null;
    public static Retrofit getClient(String baseUrl){
        HttpLoggingInterceptor interceptor = new HttpLoggingInterceptor();
        interceptor.setLevel(HttpLoggingInterceptor.Level.BODY);
        OkHttpClient client = new OkHttpClient.Builder()
                .addInterceptor(interceptor).build();
        if(retrofit == null){
            retrofit = new Retrofit.Builder()
                    .baseUrl(baseUrl)
                    .addConverterFactory(GsonConverterFactory.create())
                    .build();
        }
        return retrofit;
    }
}
```

Fig. 10. RetrofitClient class

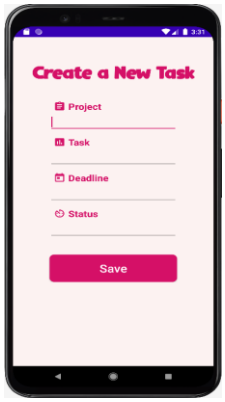

Fig. 11. Layout design for new tasks

On the server side, the researcher makes CRUD operations to handle requests from clients, in this case Android applications. One of the scripts on the server side is the create operation which is used to enter data into the database. This program code is written using the PHP programming language. The program code to perform the create operation is shown below:

```
1. <?php
2.     require_once 'connect.php';
3.     $project  = $_POST['project'];
4.     $task     = $_POST['task'];
5.     $due      = $_POST['due'];
6.     $status   = $_POST['status'];
7.
8.    $due_newformat=date('Y-m-d', strtotime($due));
9.
10.     $query = "INSERT INTO tbl_list (project,
task, due, status)
     VALUES ('$project', '$task', '$due_newformat',
'$status')";
11.     if (mysqli_query($conn, $query) ){
12.         $response["value"] = 1;
13.         $response["message"] = "Task
successfully added";
14.         echo json_encode($response);
15.        mysqli_close($conn);
16.      } else {
17.         $response["value"] = "0";
18.         $response["message"] = "Task failed to
added";
19.        echo json_encode($response);
```


Ester Lumba, "Data Exchange Using JavaScript Object Notation and REST API," *International Journal of Multidisciplinary Research and Publications (IJMRAP)*, Volume 5, Issue 3, pp. 42-47, 2022.

```
20.         mysqli_close($conn);
21.     }
22. ?>
```

Using the PHP programming language, it is possible to encode and decode JSON code with PHP's built-in build in functions, namely json_encode() and json_decode(), so you don't have to worry about encoding and decoding json data anymore. The json_encode() function returns a JSON representation of a value. In other words, it converts a PHP variable (containing an array) to JSON. In the above example the json_encode() function on line 14 is to encode the data entered from the Android application to the database and message if the data is successfully entered.

4. Testing Phase

At this stage the researchers conducted functional testing with the blackbox method. The overall test results for data input, data updating, deleting data and displaying data have been functioning properly. After conducting a series of tests, the application is then released to users or to the play store.

III. Result and Discussion

The Android application used to test the data exchange is a project monitoring application by the project manager. On the client side, the language used is Java and XML for the layout design. If run, it will produce a display like Figure 12. below:

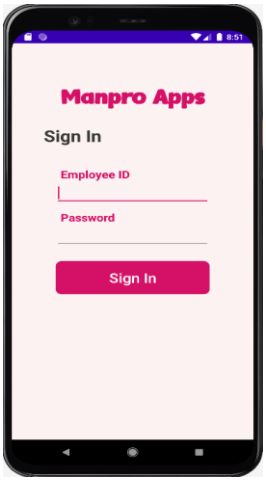

Fig. 12. Login Activity

Use the Employee ID and password to login to the Manpro application. If the employee ID and password entered are incorrect or not in the database, the login fails. However, if the employee ID and password are entered correctly, they will enter the application. After logging in, the application can display the activity menu as shown in Figure 13.

In the Activity menu, users can add new tasks by pressing the Add a list button as shown in Figure 11 above. Project managers can monitor and view a list of projects being created by pressing the List To Do button as shown in Figure 14.

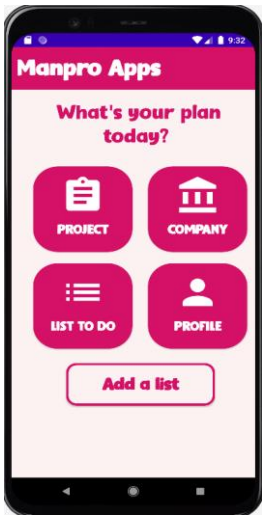

Fig. 13. Menu Activity

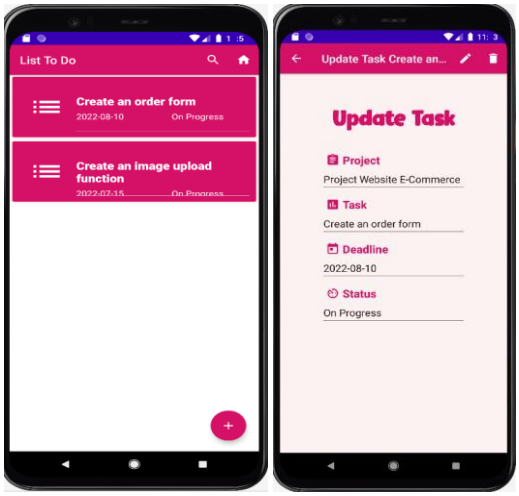

Fig. 14. List To Do and Update Task Activity

And to update the data, you can click on the project list, it will display the project details. To delete data, press the delete icon in the upper right corner.

IV. Conclusion

Using the Retrofit library can simplify the process of exchanging data from data sources namely MariaDB and client-side applications using Android. The data from the data source is converted to JSON format using the json_encode() function and the client side using the GSON library converts Java objects into JSON format.

Acknowledgment

This research was conducted independently by the researcher, to support the mobile programming teaching materials carried out by the researcher.

References

[1] S. Kemp, "DIGITAL 2022: INDONESIA," DataReportal, 15 2 2022. [Online]. Available: https://datareportal.com/reports/digital-2022-indonesia. [Accessed 20 5 2022].

[2] E. Lumba, "Pertukaran Data Pada Aplikasi Android Menggunakan Java

Script Object Notation (JSON) Dan REST API dengan Retrofit 2," in *Prosiding Seminar Nasional Aplikasi Sains & Teknologi (SNAST) 2021* , Yogyakarta, 2021.

[3] V. S. Jiri Hradil, "Practical Implementation of 10 Rules for Writing REST APIs," *Journal Of Systems Integration*, p. 45, 2017.

[4] H. Y. Jaydevi Bhade, "Evaluation of Android Networking Libraries," *International Journal of Scientific Research & Engineering Trends*, p. 1374, 2019.

[5] Developer, "Introduction JSON," [Online]. Available: https://www.json.org/json-en.html. [Accessed 14 6 2022].

[6] E. International, "ECMA-404 The JSON data interchange syntax," 12 2017. [Online]. Available: https://www.ecma-international.org/publications-and-standards/standards/ecma-404/. [Accessed 10 3 2022].

[7] F. Doglio, Pro REST API Development with Node.js, Berkely, CA, United States: Apress, 2015.

[8] V. R. Anshu Soni, "API Features Individualizing of Web Services:," *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, vol. 8, no. 9, p. 664, 2019.

[9] J. DiMarzio, Beginning Android Programming with Android Studio, Indianapolis, Indiana: John Wiley & Sons, 2017.

[10] JavaTpoint, "History of Android," 2011-2021. [Online]. Available: https://www.javatpoint.com/android-history-and-versions. [Accessed 11 3 2022].

[11] G. Developers, "Arsitektur Platform," 7 5 2020. [Online]. Available: https://developer.android.com/guide/platform. [Accessed 11 3 2022].

[12] C. A. Kent Beck, Extreme Programming Explained: Embrace Change, 2nd Edition (The XP Series), Boston: John Wait, 2005.