Two Idea Implementation of Real-Time Systems

Piotr Konczak[1], Wojciech Zabierowski[1]

[1]*Technical University of Lodz, Microelectronics and Computer Science, Wólczanska Street 221/223, B18, Łódź, Poland*

Abstract **– Currently Web portals which are providing real time services, required to implement two separate servers. One of them deals contact with the client and the other synchronizes data posted by users. Such solutions require the simultaneous work of two independent machines and write twice more code. A better solution might be a program which updates data when you try to use them.**

Keywords **– Real-Time Systems, Scattered Game.**

1. Introduction

World that we know going forward intensely, together with it are developing technologies associated with all branches of our lives. Progress aims to increase benefits, make easier the work carried out, reducing operating costs. All of this does not bypass computer technology and the Internet. Each programmer trying to make its program more efficient. They optimize the code, which will be absorbed less computer resource and do the job in shorter time.

Let's take a closer look to scattered game running in real-time. We would like to write a game, that allows a large number of users competing with each other, in a virtual kingdom. In the surrounding fantasy world, players will possess the city and hire heroes. Through the management will acquire raw materials, waging wars, develope the infrastructure and upgrade skills. Will also be able to recruit units living in the kingdom to enlarge its troops. And many more. But the most important part is how it works. All events will take place in real time. For example, the construction time of a new building may take several days.

The game will be available via any Web browser. For its implementation we will use PHP language, which allows dynamic generating of Web pages. View of the page will be describe in HTML and CSS language. To improve the performance of our application we'll take some of the tasks on the client side. For this task, will use script language JavaScript. Also we use the jQuery library, to enhance the visual experience of players. Also We will use AJAX technology to reloading only the parts of documents. This should speed up the functioning of the server which will have to generate smaller parts of the code.

2. Standard Solution

Under normal circumstances, to realize such a project should implement two separate servers (Figure 1).

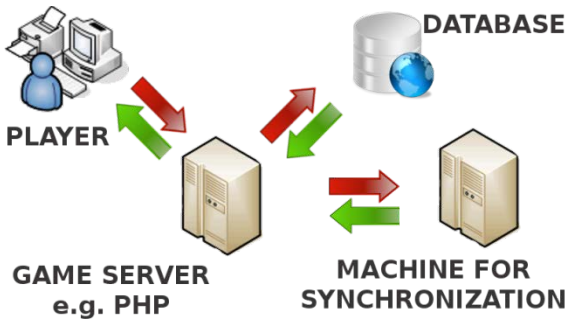

Fig. 1: Standard solution for real-time systems

Game server - its task would be communicate with the players. After receiving the request, would prepare appropriately crafted document. Inside which would be information looked by the customer. From the database would be retrieved information about the status of the player. Among the data would be found messages about activities or actions possible to taken. The customer should be able to obtain any information associated with kingdom.

Machine for synchronization – used to automatic and regular check the status of the game contained in the database. Each time, the downloaded data will be analyzed in detail. If it is noted that among them are data which term of validity has expired, to game server will be sent a request to update them.

Among the disadvantages of such a solution, we can mention:

- Need to implementation of two servers. It increases twice the amount of work for developers. While they definitely will not work for free.
- In most cases, the servers will be based on different programming languages. As a result, we will need a programmer or even more programmers familiar with both of them for system administration.
- Significantly grows the prospect of an error. When on one of the servers an error occurs continuing provision of services will become impossible. While the second server will still attempt to perform their task. This may cause completely unpredictable events of catastrophic consequences.
- In addition to having at equipment two machines, we need to maintain them. Due to the type of provided service it must be available for 24 hours a day throughout the year. Power consumption for such a solution shall be as small as possible, but it can be difficult to do.
- In specific cases desynchronization of data, misleading the customer. User read the information on the completion of certain events at a specified time, but the synchronizing machine did not work.
- When the synchronizing machine fails, users will still use the Web site. Using the data available to them, they will perform operations that may lead to a conflicting situation when the server of synchronizing will be fixed.

3. Alternative Solution

To implement such systems may be used a different approach. Must be created a system based on actions and reactions. We completely resign from the additional synchronizing machine. Updating logic put in code of the game. When a player tries to open page containing information changing in time, will be checked for their expiry date already does not expired. In the case where the data update time has passed, the updating function will be called. When it ends its operations, the data will be sent to user (Figure 2).

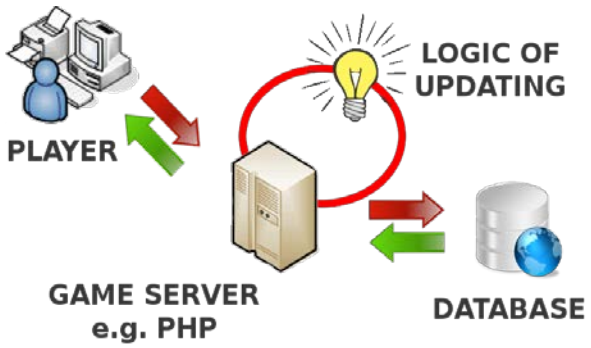

Fig. 2: Alternative solution based on appearance of user

Let us take the specific case from game. The player opens tab of the city, which belongs to him. A few hours earlier, he ordered start the construction of a new building. Information readed then, said that the work be completed until the next visit. During examination it appeared that the specified time has really passed. Updating function introduced into the database, information about recently constructed building. And from now user was able to use it.

Let us describe another one, more complicated case. Players use their heroes to fight wars among themselves. During each battle will be able to prove their strategic skills. To do this they will be able to command troops to move or attack. The action shall take troops on change in the order of possessed initiative. However we want to avoid a situation where a player due to a long absence has blocked the battle and the other players. For this purpose, we determine maximum time which will have a branch to execute the action. During the downloading information for the user, check whether the specified time has does not expired. If so, the updating function takes control of the division, made the most profitable action and give movement to next unit. It should be noted that the branch whose action was automated does not have to belong to the player retrieving the page. It is also necessary to verify whether the time since last visit is not long enough that the branch next in line should also lose the ability to perform the action.

The benefits of this solution:

- Need only one server. We save all the costs associated with the synchronizing machine. The cost of purchase, maintenance and software.
- Lower risk of error occurs. We use only one machine that can be damaged. When such event

will take place immediately know on which occurred.

- The risk of data desynchronization does not occur. If the data are already out of date will be updated. User will never get information while time has already elapsed, but he is still unable to use them.

- There is no risk that customers will use from the Web site when synchronized machine will not function. This eliminates the possibility of conflicting data.

4. Troubleshooting

Carrying out such a system, we need to define at the beginning events related the execution in real time. However, in a standard solution also must be determined. Once when define design objectives, has to be put appropriate fields holding date of expiry into the database. In addition, must be specified all of use cases where will need to update the data. It will be as simple how a situation when the user will want to use his data. But also more complex when one player will be refreshing the data associated with many others.

The data that will be currently refreshing should not be at the moment available for other users. This involves the necessity of implementation global variables informing about busy of the data. For example, the described functionality provides me the database. PostgreSQL automatically takes care of the atomicity of stored data (Figure 3).

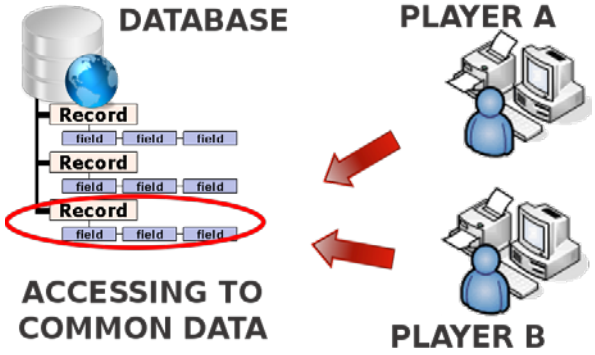

Figure. 3: Attempt of simultaneous access to shared data during the update

Operations executed recursively until get the current data, may take a long time. Such code should be treated especially, by devoting more time to its optimization. Should be also prepare a notice informing user that his request is being processed. Such a document could contain information about the approximate time of waiting.

5. Results of Work

Figures of numbers 5-9, represent results of work on example game described in Chapter . After logging a player is taken to a page summarizing the events taking place in the kingdom – Figure 5.

Figure 4. Welcome page of logged user

View of map and from its level giving orders to heroes – Figure 6.

Figure 5. Map of kingdom

Figure 7 is a schematic view collecting essential information about a hero.

Figure 7. Hero view

To develop infrastructure of the city is used tab like in the Figure 8.

Figure 8. Tab of the city infrastructure

The armed conflict will be resolved on the battlefield, like shown on Figure 9.

Figure 9. Battlefield

6. Conclusion

The presented system does not provide for the implementation of projects based on the regular collection of information from the outside. E.g. an application to get information about current weather, subscribing condition in every 15 minutes.

For web applications operating as a real-time services, we can use only one server. Will allow us to save money and time. Reduce the risk of errors and costs of their possible repair. However, requires a precise definition of use cases. And implementation data updating logic before use of them. Should also be done protection that customer does not wait infinitely, and was informed about progress. In the case of simultaneous attempts to access shared data does not perform a double updates. The logic of refreshing the data only at the time of demand, will not work for systems receiving information in regular intervals.

References

[1]. Matt Rutledge, *PHP. Programowanie gier*, MIKOM, Warszawa 2005.

[2]. Alfred V. Aho, John E. Hopcroft, Jeffrey D. Ullman, *Algorytmy i struktury danych*, Helion, Gliwice 2003.

[3]. Tomasz Starczyński, Andrzej Zoła, *PHP5. Programowanie z wykorzystaniem Symfony, CakePHP, Zend Framework*, Helion, Gliwice 2010.

[4]. Refsnes Data, Sposoby użycia różnych technologii Internetowych, pierwsza połowa 2011, [http://www.w3schools.com/].

[5]. Bruce Eckel, *Thinking in Java*, Wydanie IV, Helion, Gliwice 2006.

[6]. Kirkpatrick, Marshall , *Explaining the Real-Time Web in 100 Words or Less*. ReadWriteWeb, 2009.

[7]. Chris Newman, *PHP w mgnieniu oka*, Helion, Gliwice 2005.

[8]. George Lause, Gottfried Vossen, *Obiektowe bazy danych*, Wydawnictwo Naukowo- Techniczne, Warszawa 2000.

Corresponding author: Wojciech Zabierowski
Institution: Technical University of Lodz, Microelectronics and Computer Science, Poland
E-mail: wojtekz@dmcs.pl