International Journal of Engineering & Technology, 7 (3.27) (2018) 631-634

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

The using CASE tools in the software development life cycle

Rustam A. Burnashev[1], Arslan I. Enikeev[1]

[1]Kazan Federal University
**Corresponding author E-mail: r.burnashev@inbox.ru*

Abstract

In article we give a review of the implemented CASE tool, the essence of which is to identify the necessary system properties of the software and hardware applications to provide customers with the best option that best suits their needs. The CASE tool uses telemetry, allowing developers to remotely identify and process the required characteristics of workstations by their software and hardware parts on the network. The received characteristics about workstations allow the developer to make a decision in the choice of technologies and tools in the design and development of the future software products. These apparatuses shape an incorporated advancement condition permitting the PC helped improvement of various applications in the proper field. The coordinated condition which we consider in our report speaks to the blend of SWI-PROLOG, C# and Information Base Administration Framework (DBMS) PostgreSQL. Because of the examination we got the most fundamental and imperative qualities of workstations that influence the incorporation and usage of programming item.

Keywords: *CASE tools, C#, SWI-PROLOG and PostgreSQL database management system, software, expert system*

1. Introduction

The report presents one of the implemented CASE tools, which can act as a central link in real-time communication between the developer and the customer and will serve as a driver for quick and correct workplace automation.
The CASE tool uses telemetry, allowing developers to remotely identify and process the required characteristics of workstations by their software and hardware parts on the network.

The incorporated advancement condition incorporates:

- SWI-PROLOG
- C#
- XPCE for the illustrations part
- ODBC Driver PostgreSQL DBMS
- PostgreSQL

SWI-PROLOG is an open arrival of Prolog. Being framed from the underlying information as a chain of thinking (choice standards) from the learning base ES can-settle on a choices in one of a kind circumstances for which the calculation isn't known in promotion vance. In addition, issue arrangement is required to be done in conditions where the underlying data is deficient, questionable, and equivocal, amid subjective process evaluation [1]. PROLOG apparatuses have all the earmarks of being the most fitting to the arrangement of the previously mentioned issues.

To create designs applications, the SWI-PROLOG circulation bundle incorporates devices that empower the improvement of a graphical UI. These devices for SWI-PROLOG are given by XPCE.

2. Methods

The object of the research is the development of a specialized CASE tool using telemetry, which provides the collection and analysis of characteristics for each computer individually over a computer network, and also interconnected by one local network of computers in the enterprise.
Compatibility is the ability of different objects, hardware or software components to communicate with each other successfully. An application of telemetry allows developers in real time to obtain information about all workstations where an appropriate software product will be installed using mass media such as telephone or computer networks, optical fiber or other types of wire communication about the required values of the measured characteristics of the hardware and software.
The environment architecture (Figure 1) assumes the use of a client-server application responsible for sending, collecting and further processing data

Copyright © 2018 Authors. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

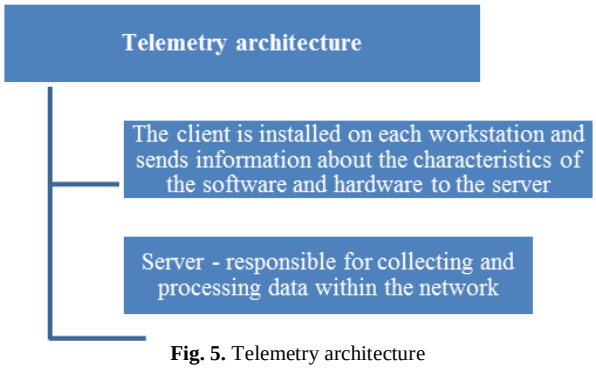

Fig. 5. Telemetry architecture

Consider the main elements of the client server application:

- Client includes:

1) Installation program
2) Configuration:
a) data for authorization
b) Name and subdivision of the organization

- Server (Fig.2 Developer Interface) includes:

1) Authorization
2) Storage of characteristics
3) Filtering requests

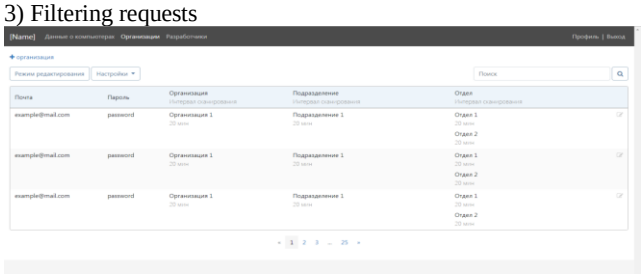

Fig.2. Developer interface

Software developers in order to achieve the set tasks are offered an extensive selection of necessary characteristics for workstations in the network.
The collection is carried out according to the following key characteristics (Fig. 7 Interface with characteristics):
1) Operating system (version, discharge, last update)
2) RAM (memory data)
3) ROM (list of available disks and memory data)
4) .NET Framework (version information)
5) Installed browsers (version information)
6) GPU (frequency, memory)
7) CPU (frequency, core, cache)
8) Sound Card
9) Network Card
10) Universally unique identifier
11) Microsoft Office (version information)
12) IP address and computer name in the network, etc.

Keeping in mind the end goal to collaborate and oversee XPCE objects from inside the SWI-PROLOG part condition, the important predicates are added to the program [2], for example,

- new (? Reference, + Class (... Arg ...))
- send (+ Reference, + Strategy (... Arg ...))
- get (+ Reference, + Strategy (... Arg ...), - Result)
- free (+ Reference)

ODBC (Open Database Network) is an application interface (Programming interface) for giving access to databases (a MICROSOFT item). So as to access to the database, it is important to choose the information source in the "Information source director" window.

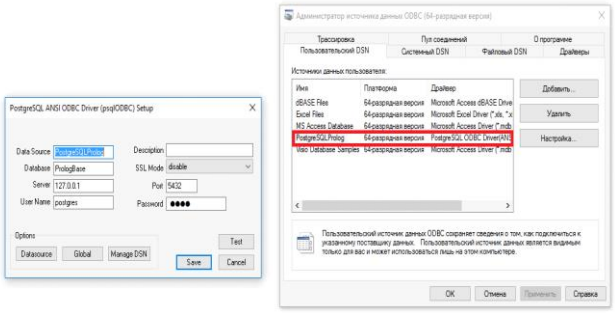

Fig. 1. ODBC Data Source Administrator.

As the aftereffect of initial step of the exploration we have built up an information base and devices for the choice of information from databases utilizing the intelligent programming dialect SWI-PROLOG and DBMS PostgreSQL.

3. Results and Discussion

The accompanying fundamental capacities were created, which empower work with PostgreSQL DBMS inside SWI-PROLOG [3]:

- Connecting to a Database Server
- Disconnecting from the database server
- Getting the rundown of tables from the associated database
- Get the rundown of segments from this table
- Retrieve all records from indicated table
- Receiving a record from current table with a predetermined UUID
- Selection of table records for a predetermined channel.

3.1 Universally Unique Identifier

A general one of a kind identifier (UUID) is a 128-piece number used to recognize data in PC frameworks. The term worldwide one of a kind identifier (GUID) is additionally utilized. At the point when made as per standard techniques, UUIDs are remarkable for pragmatic purposes, free of their uniqueness in the focal enrollment specialist or coordination between the gatherings making them, not at all like most other numbering plans. Despite the fact that the probability of copying the UUID isn't zero, it is adequately near zero to be irrelevant. Subsequently, the data assigned by the UUID by autonomous gatherings can be later consolidated into a solitary database or transmitted along a similar channel without the requirement for settling clashes between identifiers. The reception of UUID and GUID is far reaching: many processing stages bolster their age and examination of their printed portrayal.

Particularly recognizing (physical or virtual) protests by methods for a double string (called identifier) is an inherent necessity for any advanced IT-framework. Along these lines, the fundamental prerequisite for these identifiers is very self-evident: Inside the life-time of the framework, two distinct items must not hold a similar identifier. Something else, the framework won't have the capacity to recognize these articles, which may prompt framework disappointments or security breaches[9].

On the premise on directed research the accompanying GUI was created:

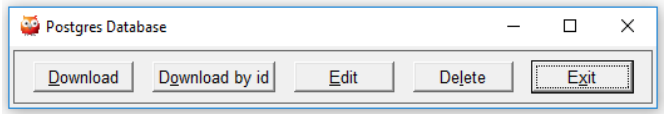

Fig.2. The menu for working with database

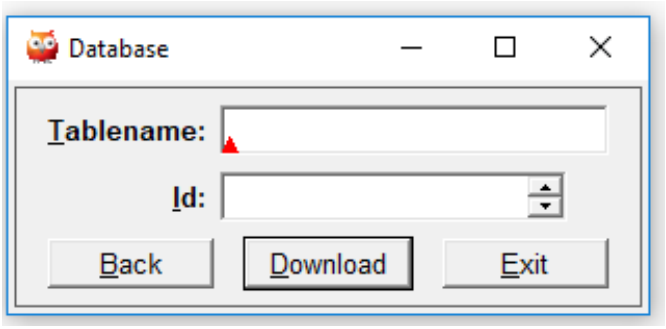

Fig.4. Downloading from database form

4. Summary

The prototype of the expert system acts as an expert, which, based on the received characteristics of the computer network, ensures the correct choice of software tools and technologies for the software developer, which will be applied to it and will ensure the correctness and quality of the software product [4].

The program characteristic is a concept of reflecting the manifestation of a separate measurable factor. In other words, a characteristic is a manifested and measurable attribute of a property. The measurement (evaluation) of one or more characteristics of the program.

Each property corresponds to one or more characteristics of the software.

To solve the problem of quantifying the characteristics of software we need a system of measurement and evaluation methods.

Quality management assumes the possibility of an independent control over the development of the PS. Control design elements are in the process of PS control.

To carry out the logical part of the study, the logical programming language SWI Prolog, the XPCE graphical library and the PostgreSQL DBMS, were used to dynamically process data on the facts of the workstation characteristics and knowledge base rules.

The prototype of the expert system acts as an expert, which, according to the characteristics of the computer network, gives an optimal choice of software tools and technologies to the software developer, who will then apply it and ensure the correctness and quality of the software. As a result of the research we received the most necessary and important characteristics of workstations that affect the integration and implementation of information systems.

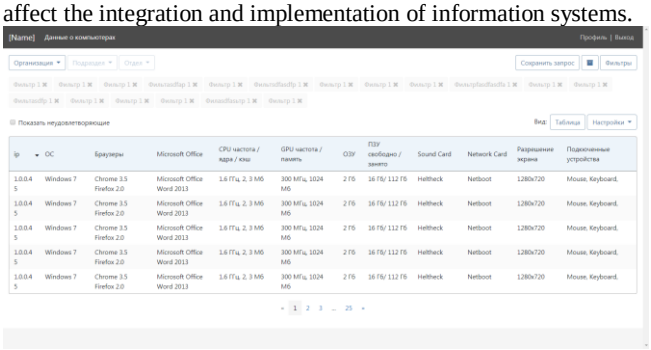

Fig. 5. Interface with characteristics

An application of specialized CASE (Computer-Aided System Engineering) tools can provide a reliability and quality of SP, since one of the main functional advantages of this technology is automation of the entire technological process of analysis, design and maintenance of software systems. Also, one of the main advantages of CASE tools is the support of the collective work of experts, engineers, subject experts, and the users themselves over the project during the life cycle of the software. In this regard, ensuring the quality and reliability of the software product is one of the most crucial problems in the field of technologies for the development of software systems.

4.1 The Algorithm Work

1. The project collects information about the software and hardware device using System.Management
2. The collected information is stored in the RAM using the project structure (classes, objects, etc.)

```
///<summary>
/// List of Central Processing Units
///</summary>
publicstaticIEnumerable<Components.CentralProcessingUnit>
CPUs
      {
get
        {
var searcher = newManagementObjectSearcher
              ("root\\CIMV2",  "SELECT  *  FROM
Win32_Processor");
foreach (varcpuinsearcher.Get())
            {
yieldreturnnewCompo-
nents.CentralProcessingUnit((ManagementObject)cpu);
}
        }
}
```

3. The information structure is serialized to the JSON format using Newtonsoft.Json, which uses reflection

```
varjson = JsonConvert.SerializeObject(new
        {
cpus = PCData.Hardware.CPUs,
gpus = PCData.Hardware.GPUs,
monitors = PCData.Hardware.Monitors,
networkadapters = PCData.Hardware.NetworkAdapters,
rammodules = PCData.Hardware.RAMModules,
romdisks = PCData.Hardware.ROMDisks,
soundadapters = PCData.Hardware.SoundAdapters,
            OS = newPCData.Software.OS(),
software = PCData.Software.All
        }, serializerSettings);
SendResponse(Client, json);
```

4. The JSON structure is sent to the server as an HTTP request (Fig.6. Software and hardware characteristics)

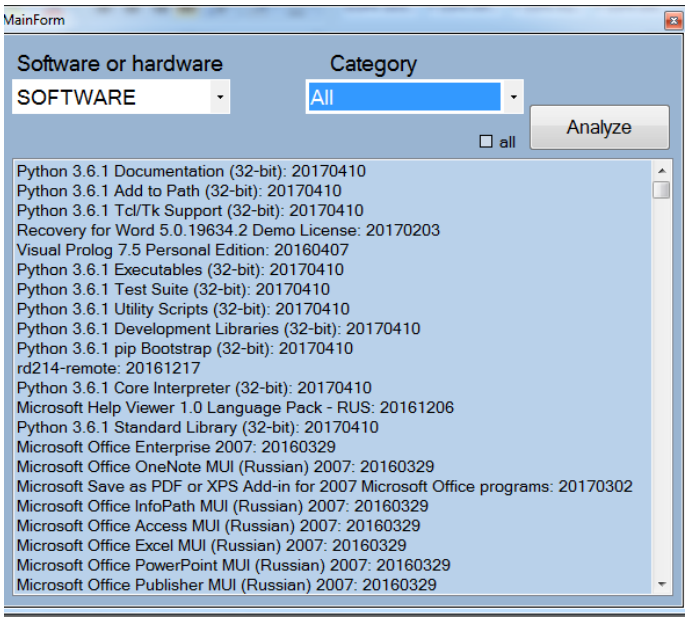

Fig.6. Software characteristics

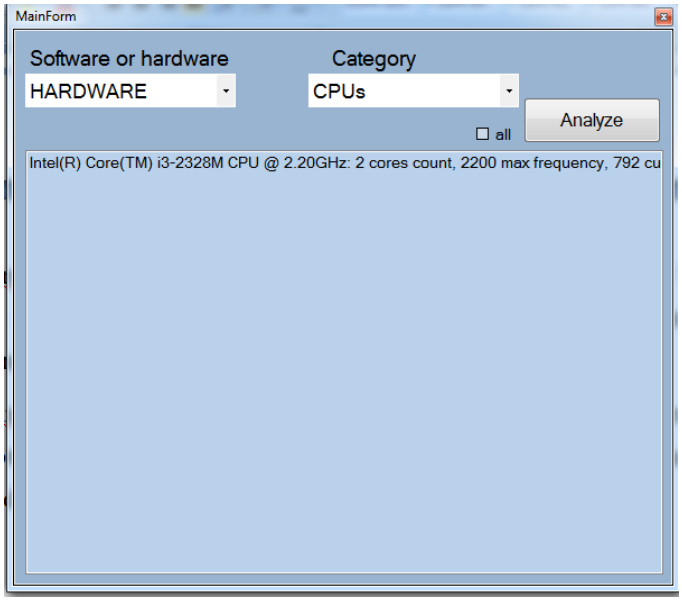

Fig.7. Hardware characteristics

5. Conclusions

The prototype of the expert system acts as an expert, which, according to the characteristics of the computer network, gives an optimal choice of software tools and technologies to the software developer, who will then apply it and ensure the correctness and quality of the software. As a result of the research we received the most necessary and important characteristics of workstations that affect the integration and implementation of information systems.

Acknowledgements

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

References

[1] P. Habarov: PROLOG – The language of Intelligent and expert systems development: a study guide. SPGLTU, Saint-Petersburg, 2013.

[2] Burnashev, R.A., Gubajdullin, A.V., Enikeev, A.I. Specialized Case Tools for the Development of Expert Systems/ Advances in Intelligent Systems and Computing/Volume 745, 2018, P.- 599-605

[3] Zainullina A. R., Larionov G. V., Burnashev R. A., Development of the Prototype Expert System Data Base Recommender//Journal Of Fundamental And Applied Sciences. - 2017. - Vol.9, Is.. - P.1417-1429.

[4] Yuriy F. Telnov: Intelligent information systems. Moscow International Institute of Econometrics, Computer Science, Finance and Right. Moscow, 2004.

[5] Hausi A. Muller, Ronald J. Norman, Jacob Slonim: Computer Aided Software Engineering, Springer US,1996

[6] Kamalov A.M, Burnashev R.A., Development of the Expert System Prototype «Medexpert» for Differential Disease Diagnostics // Astra Salvensis. – 2017. – Vol.2017, Is.. – P.55–64.

[7] Salahaldin Juba, AchimVannahme, Andrey Volkov "Learning PostgreSQL", Packt Publishing Ltd, 2015

[8] Jeremy Keith, «HTML5 for Web Designers», A Book Apart, 2015

[9] M. Schaffer, P. Schartner, and S. Rass. Universally Unique Identifiers: How to ensure Uniqueness while Preserving the Issuer's Privacy. In S. Alissi and H. R. Arabnia, editors, Proceedings of the 2007 International Conference on Security & Management – SAM'07, pages 198–204. CSREA Press, 2007.