DOI https://doi.org/10.30525/978-9934-26-008-7.2-2

EVALUATING PRODUCTIVITY OF DISTRIBUTED KNOWLEDGE-BASED SYSTEMS

Kulykovska N. A., Timenko A. V.

INTRODUCTION

To solve the problem of improving the performance of distributed systems, special systems are being developed that allow evaluating the performance of individual nodes, service composition errors, time delays, etc. Based on this data, the expert can make a decision to optimize the system. A common disadvantage of such systems is that the task of identifying performance problems remains rather difficult due to the large amount of information being processed and the complexity of the relationships that generate individual problems. Under these conditions, the task of developing a system based on the knowledge of experts to identify typical performance problems of distributed systems and ways to eliminate them is urgent.

There is a definition of knowledge based system is a computer program that operates with knowledge in a particular subject area with a view to making recommendations or solving problems. System based on knowledge, focused on a certain subject area. Such systems can serve as a complete re-placement of an expert in a given subject area or to be an intelligent assistant to the decision maker. In the latter case, the decision maker also may be an expert, and the system can improve its performance by performing rou-tine operations.

The features of knowledge-based systems that distinguish them from systems with an algorithmic approach are:

– modeling not so much the nature of a certain subject area, as the mechanism of an expert's thinking when solving problems in a given subject area;

– forming conclusions based on the knowledge that the system has;

– knowledge in systems is presented in some special language and is contained in the knowledge base of systems;

– when solving problems, heuristic and approximate methods are used, which, unlike algorithmic ones, do not always guarantee success. Systems use the knowledge accumulated by a human expert in the process of solving similar problems.

1. Development of distributed knowledge-based systems

A knowledge based system is a computer program that operates with knowledge in a specific subject area in order to make recommendations or solve problems[1].

Knowledge-based systems are focused on a specific subject area. Such systems can serve as a complete replacement for a human expert in a given subject area, or they can be an intellectual assistant for a decision-maker. In the latter case, the decision-maker can also be an expert, and the system can improve his efficiency by performing routine operations.

Knowledge-based systems are used to solve problems of interpretation, forecasting, diagnostics, design, planning, monitoring, commissioning, assistance with repairs, training and control in various problem areas[2].

The development of knowledge-based systems is possible when[3]:

– there are experts in this field who solve the problem much better than beginners;

– experts agree on the assessment of the proposed solution;

– experts are able to express in natural language and explain the methods they use;

– solving a problem requires only reasoning, not action;

– the task should not be too difficult (its solution should take an expert several hours or days, not weeks or months);

– the task should be in a fairly structured area;

– the solution of the problem should not to a large extent use "common sense" (i.e. a wide range of general information about the world and the way it functions).

The development of knowledge-based systems is justified if[4]:

– solving the problem will bring a significant effect;

---

[1] Schreiber G., Wielinga B., Breuker J. KADS : a principled approach to knowledge-based system development. London,1995. 452 p.

[2] Sakaguchi T., Matsumoto K. Development of a Knowledge Based System for Power System Restoration. IEEE Transactions on Power Apparatus and Systems. 1983. Vol. PAS-102, I. 2. P. 320–329

[3] Yang W., F, C., Yan X. et al. A knowledge-based system for quality analysis in model-based design. 2020. pp 1579–1606 DOI: 10.1007/s10845-020-01535-8.

[4] Shehab E., Abdalla An Intelligent Knowledge-Based System for Product Cost Modelling. The International Journal of Advanced Manufacturing Technology. 2020. Vol. 19. P. 49–65.

– it is impossible to use a human expert due to the limited number of experts or because of the need to carry out the examination simultaneously in several places;

– when transferring information from one expert to another, there is a significant loss of time or information;

– it is necessary to solve the problem in a hostile environment.

Of the well-known works on the application of knowledge-based systems to analyze and improve the performance of computer programs, the following can be noted:

– system provides a general overview and comparison of approaches used in expert systems to analyze the performance of computer programs[5];

– The Parallel Program Analyzer uses production rules to identify ineffective algorithms in parallel programs based on the analysis of their source code and formulate recommendations for changing the implemented algorithms[6];

– analyzer describes the use of production rules to identify the causes of insufficient performance of distributed web applications and issue recommendations for improving the program[7];

– it is proposed to use a knowledge-based system for designing real-time software systems[8];

– it is proposed to use production rules to heuristically search for the optimal parameters of the Panda parallel I / O library[9];

– systems for issuing recommendations on the choice of software libraries to achieve optimal program performance are proposed[10];

[5] Rizzo L., Longo L. An empirical evaluation of the inferential capacity of defeasible argumentation, non-monotonic fuzzy reasoning and expert systems. Expert Systems with Applications. 2020. Volume 147. P.105433. DOI: 10.1016/j.eswa.2020.113220.

[6] Li K. A knowledge-based performance tuning tool for parallel programs. Second International Conference on Algorithms and Architectures for Parallel Processing, 1996. P. 287–294.

[7] Xu J. Rule-based automatic software performance diagnosis and improvement. Performance Evaluation. 2012. Vol. 69, Issue 11. P. 525-550. DOI: 10.1016/j.peva.2009.11.003.

[8] Goettge R. T. START/ES: an expert system tool for system performance and reliability analysis. Performance Evaluation. 1995. Vol. 22, Issue 1. P. 43-58. DOI: 10.1016/0166-5316(93)E0037-6.

[9] Poyraz E., Xu H., Cui Yi. Application-specific I/O Optimizations on Petascale Supercomputers. Procedia Computer Science. 2014. Volume 29. p. 910-923. DOI: 10.1016/j.procs.2014.05.082.

[10] Houstis E. N. PYTHIA-II: a knowledge/database system for managing performance data and recommending scientific software. ACM Transactions on Mathematical Software. 2000. Vol. 26, № 2. P. 227-253. DOI: 0.1145/353474.353475.

– systems for automatic performance tuning of operating systems are proposed[11];

– a system is proposed for automatic tuning of parameters of database management systems to improve their performance[12].

The development of knowledge-based systems requires research in the field of knowledge representation models, methods of their description and interpretation algorithms. Knowledge engineering in distributed knowledge-based systems implies the use of ontological representation of knowledge. Thus, an object is formed that has a logical programmatic descriptive environment for further automatic machine processing.

The methodology for constructing the distributed knowledge-based systems is to present the semantic descriptions of all existing nodes in the system. The methodology we have developed is as follows[13]:

– each node of the system is presented as a service, i.e. a complete hardware and software object that can perform some functions;

– to introduce a new service into the system, it must be registered in it, while it must have its own semantic description (ontology);

– at this step, a node ontology is developed, which is stored in the system and is a necessary and sufficient condition for further use of the service;

– in the structure of the distributed knowledge-based systems there is a semantic service that implements the functions of searching and accumulating knowledge in the system. The search request is received at its input and is automatically rebuilt into the form of a small ontology. Further search is based on comparing the request ontology with the stored service ontologies in the system;

– the search result is constructed as follows, in the first place will be the services, the ontologies of which are maximally similar to the search queries, and then in descending order of similarity.

The semantic web should enable greater access not only to content but also to services on the web. Users and software agents should be able to discover, invoke, compose, and monitor web resources offering particular services and having particular properties, and should be able to

[11] Hoogenboom, P. J. System Performance Advisor: An Expert System for Unix System Performance Management: Master's Thesis. Salt Lake City, 1992. 144 p.

[12] Benoit, D. G. Automatic Diagnosis of Performance Problems in Database Management Systems. Proceedings of the Second International Conference on Autonomic Computing. 2005. Seattle, WA, USA. P. 326-327. DOI: 10.1109/ICAC.2005.12.

[13] Stohr, E.A., Zhao, J.L. Workflow Automation: Overview and Research Issues. Information Systems Frontiers.2001. Vol. 3(3). P. 281–296.

do so with a high degree of automation if desired. Powerful tools should be enabled by service descriptions, across the web service lifecycle[14].

The semantic web concept introduces formal definitions called ontologies, which allows you to build models of heterogeneous objects in a domain, share knowledge and support the automation of the formulation of logical conclusions from this knowledge.

To create a holistic system for deploying distributed knowledge-based systems semantic web services in a specific field of activity, it is necessary to develop a formal ontological model of objects (web services) that are part of service-oriented architecture (SOA)[15]. By combining the key elements of SOA with knowledge-based systems, you can get a formal distributed knowledge-based systems model[16].

Figure 1 shows the structural model of distributed knowledge-based systems. We single out the concept of the main artifact of the system through the Element class, its subclasses are such distributed knowledge-based systems elements as a service, system, event, human actor, semantic service[17].

The diagram shows that the service class consists of three elements: service description, application ontology, service interface[18]. The semantic principle is reflected through such elements: domain ontology, application ontology, task ontology and knowledge base[19].

---

[14] Cardoso J., Sheth. Amit P. Semantic Web Services, Processes and Applications. SWSWPC 2004: Semantic Web Services and Web Process Composition. 2004. p. 1–13.

[15] Kulykovska, N., Skrupsky, S., Diachuk, T. A model of semantic web service in a distributed computer system. CMIS-2020 Computer Modeling and Intelligent Systems: Proceedings of the Third International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020). Zaporizhzhia, Ukraine, April 27-May 1, 2020. CEUR-WS.org, online. P. 338–351.

[16] Kulykovska N.A, Timenko A.V.: A Structure of Semantic Service in a Distributed Knowledge Based System. Computer Modeling and Intelligent Systems: Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019). In: CEUR Workshop Proceedings 2353, 2019. P. 533–544.

[17] Kirichek G., System for detecting network anomalies using a hybrid of an uncontrolled and controlled neural network. CS&SE@SW 2019: 2nd Student Workshop on Computer Science and Software Engineering, Kryvyi Rih, Ukraine, November 29, 2019 P. 138–148.

[18] Allemang D. Semantic web for the working ontologist. Modeling in RDF, RDFS and OWL. Morgan Kaufmann Publishers. 2011. 348 p.

[19] Yu, Liyang. Introduction to Semantic Web and Semantic Web services. Springer. 2007. 335 p.

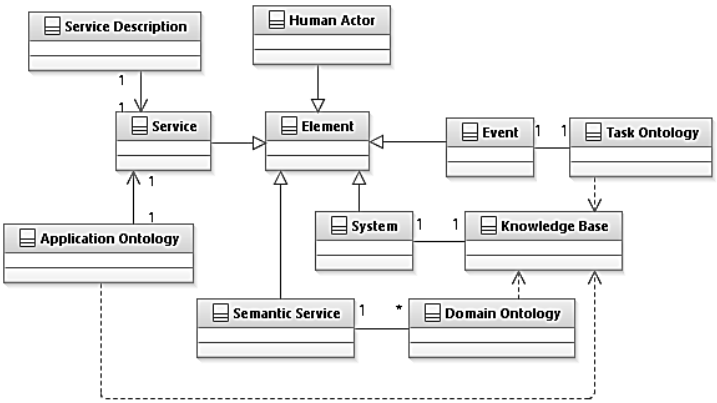

Fig. 1. Structural model of DKBS

A formal semantic service model should include:

1. Analysis of the structure of knowledge of the subject area of the distributed knowledge-based systems, the main objects and relations:

(a) the use of least-logical and ontological methods for the formation of the terminology of the system domain and knowledge structure;

(b) structure of OWL ontology.

2. Repeated use of existing thesauruses, taxonomies, and ontologies of the distributed knowledge-based systems domain:

(c) semantic search for relevant objects and analysis of means of representing knowledge and standards in the distributed knowledge-based systems;

(d) a brief overview of relevant ontologies and other knowledge structures;

(e) integration of existing taxonomies and domain ontologies.

3. The architecture of methods for the automated improvement of the formal ontological model of the distributed knowledge-based systems:

(f) the architecture of methods for the automated extraction of knowledge (terms and relationships) from natural language texts that relate to the distributed knowledge-based systems;

(g) methods of automated linguistic processing of natural language texts;

(h) advanced OWL ontology.

4. Semantic search in the system based on the domain ontology:

(i) semantic search for objects;

(j) methods of semantic search for objects of the distributed knowledge-based systems;

(k) methods of semantic search for SOA services;

(l) recommendations regarding the use of web services.

Search data for web services, their interactions, reviews, recommendations from service customers can be analyzed and converted into active knowledge, which allows us to better understand the physical world and create more value-added products and services.

2. Performance of distributed knowledge-based systems

Existing systems can be divided into two groups according to performance objectives. Systems in the first group collect and visualize performance data. The rest of the tasks (identifying performance problems and finding ways to fix them) is the responsibility of the user. When using graphical tools, the user faces the following difficulties:

1) analysis of a large amount of information and the complexity of the interaction of processes. Even with a fairly short program runtime and a small number of processes, a very large amount of information is displayed on the timing diagram and other visual tools;

2) typical performance problems (eg late message reception) are not explicitly indicated in visual aids. Thus, manual analysis of the data using the graphical tools listed above is required to determine the reasons for the lack of performance[20];

3) there is no means to issue recommendations to the user to improve performance. Optimizing performance requires deep knowledge from the user in the subject area, features of data transmission networks and other knowledge.

The second group includes relatively recently developed systems that allow solving problems of identifying performance problems and finding ways to fix them in an automatic mode. With these systems, the task of increasing productivity is greatly simplified. The disadvantages of such systems are:

– there is no formal definition of the performance problem in the system publications;

– descriptions of performance problems and how to fix them in these fixed systems, namely:

[20] Knowledge Specification for Automatic Performance Analysis. T. Fahringer [et al.] Technical Report, FZJ-ZAM-IB-2001-08, Germany, 2001. 83 p.

1) to describe performance problems, program code in general-purpose languages such as C ++ and Java is used. To expand the list of recognized performance problems, you need to change the code of these systems;

2) all considered systems operate with a fixed track format. If it is necessary to expand the list of recognized performance problems, in some cases it may be necessary to expand the trace format and, as a result, reprogram the tracer and the trace file readers and writers.

So, the descriptions of performance problems and how to fix them in existing systems are fixed. As a result, performance issues identified by an expert and decisions made to resolve them are not saved in such systems.

The proposed distributed system can solve performance problems. The developed structure allows you to save all the results of the work of services and the system as a whole. Thanks to the functions of the semantic service, you can eliminate system performance problems, create a knowledge base without having deep knowledge in the subject area. In the distributed knowledge-base system, the communication and collaboration mechanism between services is a problem that must be well considered in the design of the semantic service structure. The communication and collaboration mechanism covers the message transfer between services, information exchange between semantic service and distributed knowledge bases, and the semantic service behaviors in the entire process from request submission, request handle, to result return. These procedures can be summarized as follows:

– semantic service deploys appropriate services on its local Knowledge Base, performs the registration on the knowledge bases;

– the retrieval request accesses the system periodically. Semantic service gets retrieval requests and searches its Knowledge Base;

– semantic service returns the retrieval result to the system after the search finishes;

– the semantic service implies a classification and a sort order to the retrieval result according to its corresponding retrieval request.

User views the retrieval results on the interface and selects the needed Knowledge. With the information about the Knowledge Base embedded in the retrieval result, the user then contacts the relevant enterprise to obtain the Knowledge.

The principle of operation of the semantic service is illustrated in Fig. 2. The knowledge base is formed from three types of ontologies. The rules for recognizing performance problems are converted into ontology and the system knowledge base is placed. The working memory of a semantic service contains facts that correspond to simple events,

composite events, and identified problems and recommendations. The algorithm for using semantic service to perform an analysis is as follows. The initial data is the sequence of events of the system, let's call it the trace. Trace contains simple events *E*. The algorithm processes the events of the route in the order of the time of their occurrence and performs the following cycle:

1. Read the next event from the trace *E*.

2. Convert it to the concept of ontology and add it to the working memory of the SS.

3. Start the semantic service. If the new event is a search query, then pick up the relevant answers in the knowledge base. For an indefinite concept of a simple event, the output engine consistently executes the rules for constructing composite events, for constructed fact-composite events, the rules for identifying performance problems. The result of triggering a semantic service can be:

(a) adding a new fact-simple event to some of the constructed fact-composite events;

(b) identification of fact-composite events of working memory for compliance or non-compliance with the performance problem;

(c) delete or save a new event in the working memory;

(d) search for a saved event.

4. Retrieve the search results from the working memory or offer recommendations to solve the problem.

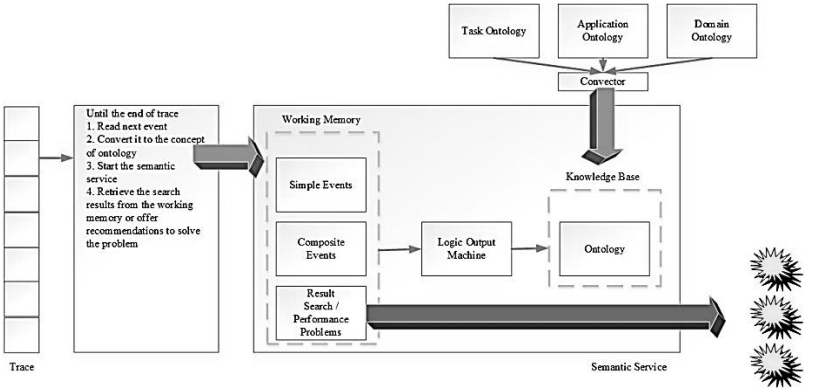

Fig. 2. Structure of Semantic Service

To assess the performance of the semantic service attached, create an information system of two components: event trace generator distributed knowledge-base system; event trace analyzer for semantic service.

Events are recorded in the track when calling action functions. A simple event is represented as:

$$e = \langle f, et, EPval, t, d, cs \rangle, \quad (1)$$

where $f \in F$ – action function;

et – the type of event that sets its parameters $et \Rightarrow EP = (ep_1, ..., ep_K)$;

$EPval = (v_1, ..., v_K)$ – event parameter values;

t – time of the event;

d – event duration;

cs – service code.

3. Evaluating the performance of distributed knowledge-based systems

We – will consider increasing the productivity of the distributed knowledge-base system as reducing the time of action within the system. Each process action a_{ij} corresponds to the beginning of its execution t_{ij}^a and duration d_{ij}^a:

$$a_{ij} \Rightarrow \langle t_{ij}^a, d_{ij}^a \rangle; \; i = 1, 2, ..., N; \; j = 1, 2, ..., M. \quad (2)$$

The loss of productivity of the interaction of web services is defined as the sum of the duration of the actions of all processes:

$$D^a = \sum_{i=1}^{N} \sum_{j=1}^{M} d_{ij}^a. \quad (3)$$

The duration of a call to a web service consists of the following components:

$$d_{ij}^a = (d_{ij}^a)_{seach} + (d_{ij}^a)_{prep} + (d_{ij}^a)_{wait} + (d_{ij}^a)_{comm}, \quad (4)$$

where $(d_{ij}^a)_{seach}$ is the duration of the search for the desired service;

$(d_{ij}^a)_{prep}$ – the duration of the preparation of data for transmission / reception of messages;

$(d_{ij}^a)_{wait}$ – the duration of the wait for the readiness of the web service;

$(d_{ij}^a)_{comm}$ – the duration of the transmission/reception of messages over the network.

Ways to reduce the duration of actions can be:

– reduction of time spent on $(d_{ij}^a)_{seach}$: the use of improved methods for finding web services;

– reduction of duration $(d_{ij}^a)_{prep}$: grouping of individual messages into a single whole;

– reduced time $(d_{ij}^a)_{wait}$: improved messaging synchronization;

– time reduction $(d_{ij}^a)_{comm}$: due to the hardware parameters of the system.

The examples of ways to reduce the time taken to complete the actions can be presented in the form of recommendations:

$$REC = \{rec_i\}, \quad i = 1, 2, ..., R. \tag{5}$$

Improving the productivity of the interaction of distributed knowledge-base system processes is defined as:

$$D^a \Rightarrow \min_{REC}. \tag{6}$$

The formulated task of increasing productivity is quite complex and cannot be solved by formal methods. Therefore, an expert methodology is proposed, with including the following steps:

1. Systematization of typical performance problems and determination of the causes of their occurrence.

2. Development of recommendations to address these causes.

3. Description of the identified performance problems, the rules for establishing the causes of their occurrence and recommendations for their elimination in the knowledge base.

The following model of the web services performance problem is applied in the work:

$$pb = \left\langle pd, dur, TrRulesAnRules, REC(A^{INFO)} \right\rangle, \tag{7}$$

where pd is the verbal description of the problem;

dur – the duration of the appearance of this problem;

$TrRules$ – rules for tracing the action of the problem;

$AnRules$ – rules for recognizing problems;

REC – recommendations for its elimination;

$A^{INFO} = \left\{\left\langle f, t, d, pr_i, cs_i \right\rangle\right\}$ – a description of the actions leading to the problem.

The rule for identifying a performance problem is represented as:

$$ce = \langle cet,\ CEPval,\ ME \rangle \xrightarrow{pbrule} pb = \left\langle pd,\ dur,\ REC\left(A^{INFO}\right)\right\rangle,$$

$$pbrule = \langle cet,\ pd,\ RECtemp,\ L_{dur} \rangle, \quad (8)$$

where cet is the type of the composite event as a candidate for the identified performance problem;

$CEPval$ – condition for the occurrence of the problem (if the value of the function is 1, then the problem exists and not otherwise);

pb – verbal description of the problem defined by the rule;

$RECtemp : CEPval \times REC$ – recommendation template for troubleshooting;

L_{dur} – function for calculating the duration of a performance problem.

When the conditions of the rule are met, a conclusion is made that there is performance problem pb. The software recursively extracts simple events from the route and generates information about a variety of actions based on them.

The rules for identifying performance problems are described using a sequence of ontological axioms with the following syntax:

declare problem for <event name>
when
<logical expression>
parameters (
name = <problem name>,
description = <problem description>,
advice = <commissions>,
duration = <expression for calculating the duration>);

where *declare problem f*or is a description of the rule for identifying a performance problem for an applicant event cet;

when – condition for the occurrence of the problem (as a logical expression over the parameters of a composite event);

name – name of the problem;

description – verbal description of the pb problem;

advice – recommendation template for its resolution $RECtemp$;

duration is an expression for calculating the duration of the manifestation of this problem L_{dur}.

Events are recorded in the track when calling action functions. A simple event is represented as:

$$e = \langle f, et, EPval, t, d, cs \rangle, \quad (9)$$

where $f \in F$ is the action function;

et – the type of event that sets its parameters $et \Rightarrow EP = (ep_1, \ \ldots, \ ep_K)$;

$EPval = (v_1, \ \ldots, \ v_K)$ – values of the event parameters;

t – time of the event;

d – duration of the event;

cs – service code.

A software environment was developed for working with ontologies and a knowledge base. The knowledge base of the system included descriptions of the following typical performance problems of distributed knowledge-base system.

The problem of knowledge engineering:

1) service ontology error;
2) search error;

Bandwidth issue:

1) the limit of service operations has been exceeded;
2) the time of using the service has been exceeded.

Concurrency problem:

1) late sending of the message;
2) late message reception;
3) synchronization error;
4) memory access delays.

Failure problem:

1) errors in the service;
2) errors in incoming parameters;
3) network / port errors.

A knowledge base has been developed for identifying typical problems of service performance, described using the proposed models and methods. The results of the study of the developed models, methods and tools in the analysis of semantic service performance have shown their effectiveness.

CONCLUSIONS

Modern representations of data are changing the ways and forms of communication, production and consumption of information. The dominance of horizontal relations, structure-role of information decentralization of all types of data available at any time on every device. The user should not care about the specific technology used to provide computing capacity or data storage, so you can say that a user has some information about the remote resource. Distributed knowledge-base system in order to study the data, their processing and use evolving technologies of the semantic web. The formal model of the distributed

knowledge-base system consists of a set of ontologies; lots of services; a set of events that describe the processes of the system; semantic service, a set of composite services and knowledge base. The completeness and effectiveness of the system is determined by a multitude of ontologies. The main difference in distributed knowledge-base system is the use of the service approach and ontologies in knowledge engineering. SS acts as a software agent that regulates the interaction of all components of the system. Semantic service provides: semantic principle, which defines a formal description of information and allows you to define the following characteristics of services: scalability, semantic interoperability, formal models of services and ontologies; the principle of decision making; the principle of distribution, which allows you to aggregate the capabilities of several computing objects through cooperation. The communication and collaboration mechanism of the distributed knowledge-base system covers the message transfer between services, information exchange between semantic service and distributed knowledge bases, and the semantic service behaviors in the entire process from request submission, request handle, to result return.

The proposed methodology for evaluating productivity of distributed knowledge-base system is to provide semantic descriptions of all existing nodes in the system. Ontologies of subject areas to develop systems that can accumulate the knowledge of experts in identifying typical performance problems of system and how to solve them. A knowledge base has been developed for identifying typical problems of service performance, described using the proposed models and methods.

SUMMARY

To solve the problem of improving the productivity of distributed computer systems, special systems are being developed to assess the performance of individual nodes, errors in the composition of services, time delays, etc. The expert can decide to optimize the system based on these data. This paper proposes the use of knowledge engineering tools in systems development. The proposed construction methodology is to present semantic descriptions of all existing nodes in the system. Service ontologies allow you to develop systems that can accumulate expert knowledge to identify common system performance issues and ways to address them. The knowledge base for identification of typical problems of productivity of services, described with use of the offered models and methods is developed.

REFERENCES

1. Schreiber G., Wielinga B., Breuker J. KADS : a principled approach to knowledge-based system development. London, 1995. 452 p.
2. Sakaguchi T., Matsumoto K. Development of a Knowledge Based System for Power System Restoration. *IEEE Transactions on Power Apparatus and Systems*. 1983. Vol. PAS-102, I. 2. P. 320–329.
3. Yang W., Fu C., Yan X. et al. A knowledge-based system for quality analysis in model-based design. 2020. Pp. 1579–1606 DOI: 10.1007/s10845-020-01535-8.
4. Shehab E., Abdalla An Intelligent Knowledge-Based System for Product Cost Modelling. *The International Journal of Advanced Manufacturing Technology*. 2020. Vol. 19. P. 49–65
5. Rizzo L., Longo L. An empirical evaluation of the inferential capacity of defeasible argumentation, non-monotonic fuzzy reasoning and expert systems. *Expert Systems with Applicatio*ns. 2020. Volume 147. P. 105433. DOI: 10.1016/j.eswa.2020.113220.
6. Li K. A knowledge-based performance tuning tool for parallel programs. *Second International Conference on Algorithms and Architectures for Parallel Processing*. 1996. P. 287–294.
7. Xu J. Rule-based automatic software performance diagnosis and improvement. *Performance Evaluation*. 2012. Vol. 69, Issue 11. P. 525-550. DOI: 10.1016/j.peva.2009.11.003.
8. Goettge R. T. START/ES: an expert system tool for system performance and reliability analysis. *Performance Evaluation*. 1995. Vol. 22, Issue 1. P. 43–58. DOI: 10.1016/0166-5316(93)E0037-6.
9. Poyraz E., Xu H., Cui Yi. Application-specific I/O Optimizations on Petascale Supercomputers. *Procedia Computer Science*. 2014. Volume 29. p. 910–923. DOI: 10.1016/j.procs.2014.05.082.
10. Houstis E. N. PYTHIA-II: a knowledge/database system for managing performance data and recommending scientific software. *ACM Transactions on Mathematical Software*. 2000. Vol. 26, № 2. P. 227–253. DOI: 0.1145/353474.353475.
11. Hoogenboom P.J. System Performance Advisor: An Expert System for Unix System Performance Management: Master's Thesis. Salt Lake City, 1992. 144 p.
12. Benoit D.G. Automatic Diagnosis of Performance Problems in Database Management Systems. Proceedings of the Second International Conference on Autonomic Computing. 2005. Seattle, WA, USA. P. 326–327. DOI: 10.1109/ICAC.2005.12.
13. Stohr, E.A., Zhao, J.L. Workflow Automation: Overview and Research Issues. *Information Systems Frontiers.*2001. Vol. 3(3). P. 281–296.

14. Cardoso J., Sheth. Amit P. Semantic Web Services, Processes and Applications. *SWSWPC 2004: Semantic Web Services and Web Process Composition.* 2004. p. 1–13.
15. Kulykovska, N., Skrupsky, S., Diachuk, T. A model of semantic web service in a distributed computer system. *CMIS-2020 Computer Modeling and Intelligent Systems: Proceedings of the Third International Workshop on Computer Modeling and Intelligent Systems (CMIS-2020).* Zaporizhzhia, Ukraine, April 27-May 1, 2020. CEUR-WS.org, online. P. 338–351.
16. Kulykovska N.A, Timenko A.V.: A Structure of Semantic Service in a Distributed Knowledge Based System. *Computer Modeling and Intelligent Systems: Proceedings of the Second International Workshop on Computer Modeling and Intelligent Systems (CMIS-2019).* In: CEUR Workshop Proceedings 2353, 2019. P. 533–544.
17. Kirichek G., System for detecting network anomalies using a hybrid of an uncontrolled and controlled neural network. *CS&SE@SW 2019: 2nd Student Workshop on Computer Science and Software Engineering*, Kryvyi Rih, Ukraine, November 29, 2019. P. 138–148.
18. Allemang D. Semantic web for the working ontologist. Modeling in RDF, RDFS and OWL. *Morgan Kaufmann Publishers*. 2011. 348 p.
19. Yu, Liyang. Introduction to Semantic Web and Semantic Web services. *Springer*. 2007. 335 p.
20. Knowledge Specification for Automatic Performance Analysis. T. Fahringer [et al.] Technical Report, FZJ-ZAM-IB-2001-08, Germany, 2001. 83 p.

Information about authors:
Kulykovska N. A.,
Assistant at the Department of Computer Systems and Networks
"Zaporizhzhia Polytechnic" National University
64, Zhukovsky str., Zaporizhzhia, 69063, Ukraine

Timenko A. V.,
Assistant at the Department of Computer Systems and Networks
"Zaporizhzhia Polytechnic" National University
64, Zhukovsky str., Zaporizhzhia, 69063, Ukraine