УДК 004.453
DOI: 10.25559/SITITO.14.201802.317-324

ADAPTATION OF SOFTWARE SYSTEMS TO USER TASKS BASED ON THE METHOD OF CONFIGURATION MODELING

Valentin V. Nechaev, Anton S. Bashirov, Natalia I. Lebedeva, Mikhail A. Fedin
MIREA – Russian Technological University, Moscow, Russia

АДАПТАЦИЯ ПРОГРАММНЫХ СИСТЕМ К ЗАДАЧАМ ПОЛЬЗОВАТЕЛЕЙ НА ОСНОВЕ МЕТОДА КОНФИГУРАЦИОННОГО МОДЕЛИРОВАНИЯ

В.В. Нечаев, А.С. Баширов, Н.И. Лебедева, М.А. Федин
МИРЭА – Российский технологический университет, г. Москва, Россия

Keywords

Adaptation; configuration modeling; conceptual modeling; system modeling; software system; configuration.

Abstract

In this paper, we discuss task of adaptation of software systems (PCs) and complexes (PCs) on the basis of the configuration modeling (MCM) method is considered. The article describes designated purpose of MCM in the field of SSs development, describes background and justification for the development of adaptive SSs, and many of the advantages of creating configurable SSs. The main attention is paid to the analysis and research of the software engineering and implementation of adaptive SSs using the configuration mechanism as a mean of adapting SSs to the problem of interest. The potential for using configuration modeling in this area is determined. The meaning of the concepts «configurator», «configuration» and «reconfiguration» in the context of the description of MCM implementation of adaptive SSs creation is considered. The article gives a real example of an adaptive SS describing MCMs application in the process of software engineering. A conceptual model of such a SS is described, and its interpretation is implemented in the form of a modular complex data processing system (CDPS). The example of the CDPS shows the structural scheme of the conceptual model of the data handling process, decomposes the CDPS into the basic modules, including the built-in configuration module. Special attention to a detailed description of the problem of interest and the functionality of each module is paid, as well as to the description of the relationships between them. On the basis of the CDPS, the solution of the problem of adapting PSs with the elements of dynamic configuration based on the input data is realized. The example of the CDPS describes user-accessible operating modes of the SS corresponding to various system configurations determined according to the tasks to be solved. The article makes it possible to obtain an abstract presentation on the methods and principles for the creation of adaptive SS. Moreover, a concrete understanding of the implementation of adaptation by using the configuration management mechanism is covered. Additionally, the article highlights direct benefits and potential profit from using of configurator in software engineering.

Ключевые слова

Адаптация; конфигурационное моделирование; концептуальное моделирование; моделирование систем; программная система; конфигурация.

Аннотация

В данной работе рассматривается задача адаптации программных систем (ПС) и комплексов (ПК) на основе метода конфигурационного моделирования (МКМ). Статья детально рассматривает предназначение и целевое применение МКМ в области разработки ПС, приводятся предпосылки и обоснование актуальности разработки адаптивных ПС, а также описывается ряд основных достоинств создания конфигурируемых ПС. Основное внимание уделяется анализу и исследованию вопроса проектирования и реализации адаптивных ПС с использованием механизма конфигурирования как средства адаптации ПС к решаемым задачам, определяется потенциал использования

About the authors:

Valentin V. Nechaev, Candidate of Technical Sciences, Professor, Professor of the Department of Instrumental and Applied Software, MIREA – Russian Technological University (78 Vernadsky Prospekt, Moscow 119454, Russia), ORCID: http://orcid.org/0000-0001-7171-3874, nechaev@mirea.ru

Anton.S. Bashirov, student of the Department of Instrumental and Applied Software, MIREA – Russian Technological University (78 Vernadsky Prospekt, Moscow 119454, Russia), ORCID: http://orcid.org/0000-0002-4113-0704, antonbashir@mail.ru

Natalia I. Lebedeva, student of the Department of Instrumental and Applied Software, MIREA – Russian Technological University (78 Vernadsky Prospekt, Moscow 119454, Russia), ORCID: http://orcid.org/0000-0003-3421-4923, natalia.i.lebedeva@yandex.ru

Mikhail A. Fedin, student of the Department of Instrumental and Applied Software, MIREA – Russian Technological University (78 Vernadsky Prospekt, Moscow 119454, Russia), ORCID: http://orcid.org/0000-0001-9398-8297, vbif.32@ya.ru

конфигурационного моделирования в данной сфере. Рассматривается смысл понятий «конфигуратор», «конфигурирование» и «реконфигурирование» в контексте описания применения МКМ к созданию адаптивных ПС. В статье приводится реальный пример адаптивной ПС с описанием применения МКМ в процессе её проектирования. Описывается концептуальная модель такой ПС и осуществляется её интерпретация в форме модульной комплексной системы обработки данных (КСОД). На примере КСОД приводится и рассматривается структурная схема концептуальной модели процесса обработки данных, представляется декомпозиция КСОД на базовые модули, включая встроенный модуль-конфигуратор. Отдельное внимание уделяется подробному описанию решаемых задач и функциональных возможностей каждого отдельно взятого модуля, а также описанию взаимосвязей между ними. На основе КСОД реализуется решение задачи адаптации ПС с элементами динамического конфигурирования по входным данным. В процессе рассмотрения применения МКМ на примере КСОД приводится описание нескольких доступных пользователю режимов работы ПС, соответствующих различным конфигурациям системы, определяемых согласно решаемым задачам. Статья позволяет получить как абстрактное представление по методикам и принципам создания адаптивных ПС, так и конкретное понимание особенностей реализации адаптации путём использования встроенного в ПС механизма управления конфигурациями. Дополнительно, в статье выделяется непосредственная польза и потенциальная прибыль от внедрения конфигуратора при проектировании ПС.

Introduction

Modern large software systems (P-systems) are developed and maintained in conditions of constantly evolving requirements. Changes in existing P-systems are associated with additional costs, increasing with the passage of time. It is the desire to reduce the volume of such changes which was the main reason for creating adaptive software (AS) systems. AS-systems should have the property of purposeful development under the influence of external conditions. At the same time, the goal of the AS system should be achieved, despite the changes. To ensure compliance with these requirements, the most rational is the modular architecture of the AS system [1]. It makes it possible to easily exchange data with the serviced system [2]. Adaptation acts as an object management tool in the absence of its exact model. Adaptation acts as an object management tool in the absence of its exact model. Adaptation can be implemented through the use of parametric, organizational, structural methods, as well as evolutionary algorithms [3,4]. To solve the problem of adapting the software system the authors of this paper use the method of configuration modeling [5]. The configuration is a qualitative characteristic of the structure that determines its spatial, logical, temporal, and also combined - complex - organization of the P-system. From the conceptual point of view, a configuration is a structure of structures or a meta-structure. Configuration modeling, as a method of adapting modular P-systems, makes it possible to significantly expand their functionality and reduce additional resources in the development of software products. The P-system can be defined as configurable if it can be configured without programming additional functions and / or without changing the source code of the program. The presence of configuration mechanisms in the AS-system provides flexibility in the use of software within the scope of the tasks and the existing environment. It should be noted that the configuration method was initially used to form configurations of computing systems, complexes and networks [4]. However, it can be relatively easily used also for P-systems. It should be noted that the ability to configure software products has existed for a long time. However, the creation of P-systems, the structure of which was initially based on broad possibilities for adaptation in the presence of certain goals, was used relatively recently.

1. The objective of the work

The objective of the work is to consider the features of the configuration modeling method applied to flexible software systems, the use of this method as a tool for adapting such P-systems to user-specific tasks, as well as demonstrating the applicability of the configuration mechanism by creating an author's P-system.

The article includes: an analysis of the relevance of the issue of creating a demonstrated P-system; description of adaptation of P-systems based on configuration modeling; a conceptual model of the adaptive P-system; implementation of the AS-system using the example of an integrated data processing system (IDPS).

2. The urgency of developing adaptive software systems

Modern P-systems are widely used in almost all spheres of society. The needs of users are constantly growing. The functionality that the P-system performed yesterday may now be insufficient or not relevant. Thus, there is a need to update the already commissioned P-system. It should be emphasized that the volumes of the source codes of modern P-systems reach the sizes at which when summing the entire program code of the average P-system in one file, it can not be supported. Eventually, this situation leads to an increase in the complexity of the distribution of the functional of the P-system in accordance with the needs of users [6].

3. The method of configuration modeling and the problem of adaptation of P-systems

To solve the task in this paper, the method of structural adaptation is used. The P-system is implemented on the basis of a modular approach. For adaptation the configuration modeling method [1] is used. To concretize, as an example, two adaptation tasks are defined. The first is the task of updating an existing active P-system. The second is the problem of decomposition of the active functional of the P-system in accordance with the needs of the user. Adaptation is achieved through the execution of the configuration mechanism [7]. The configuration mechanism is to reform the structure of the P-system by changing the current set of its modules and their interrelationships. The configuration management of the P-system is provided through a separate software module - the configurator. The configurator (K) is a component of the P-system, the purpose of which is the purposeful configuration and reconfiguration of the main part of the P-system. It is assumed that the P-system can be in different states and has several different operating modes. The mode of operation is understood as the cumulative cross-section of the current parameters of the P-system that determine its state, as well as sets of tasks that the system solves while in any mode.

In the role of the configurator (we denote it by the symbol K) there can be an external program with respect to the P-system, operating in the same environment as the configurable one. However, it should be taken into account that the presence of the K configurator as a separate component from the P-system of the software component, poses the challenge of creating a control channel that would provide lossless transfer of control information from the configurator K to the configurable P-system, as well as information (demand) from the P-system to the K configurator, see Figure 1. Using the modular architecture of the P-system in combination with the K configurator allows performing partial updates of the P-system within a limited number of modules, without affecting other parts of the system. This solves the above problem of adapting the P-system for updating its active copy [1].

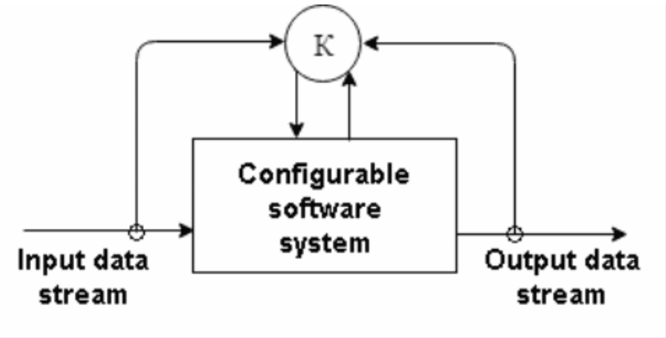

Fig. 1. Schematic diagram of the PS configuration management control

Рис. 1. Схема ПС управления конфигурациями

The task of decomposition of the P-system functional in accordance with the user's needs is solved through the use of configurable modes of its operation. In other words, using the configurator K, we can change the current configuration of the modules according to the user's needs at a given time or, for security reasons, limiting the user-accessible area of the system functionality to the dedicated set of modules. Under changing requirements for the system, the merits of using this approach are:

- no need to make improvements to an existing system, which can lead to an increase in the cost of the project and the timing of its implementation;
- the possibility of making changes to the functionality of an existing system by highlighting the modes of its operation, determined by the structure and composition of the interacting modules without modifying the modules themselves;3
- no need to create a system from scratch to meet new requirements.

As an example of the practical implementation of the adaptation task through configuration modeling, let us consider an integrated data processing system.

4. Integrated data processing system

Integrated data processing system with the provision of a calculation mechanism consists of two subsystems: client file processing (KO) and server processing of files with a computational mechanism (CO). The advantage of creating and using a configurable IDPS is determined by the fact that it has a number of properties, among which we distinguish the following:

- the possibility of step-by-step commissioning of a separate functional that implements a certain range of automation tasks for business processes;
- automation of a long process of data calculation;
- reduction of the probability of human errors in the calculation;
- saving resources by designing a rational system configuration.

The structural scheme of the data processing process is shown in Figure 2.

5. Structural diagram of the conceptual model of the data processing system

The conceptual model of the system in the narrow sense is the collection of components of the system and their interactions. The process of data processing can be considered in two ways: as a process consisting of several actions; a program as a set of components, where the component is understood as the program module executing the processing of data.

From the process approach point of view, data processing consists of the following set of actions: data aggregation, data conversion, data storage, data usage, generation of results of data use.

The process of data processing can be repeated until the end result satisfies the user's needs. In this case, each subsequent iteration of the process will be applied to the result of using or converting data of the previous one.

The component scheme of the data processing process is as follows:
DH = <D, A, C, S, U, R>, where
D is the set of data sources
A - data aggregator
C - set of data converters
S - database
U is the set of data users
R is the set of results of data usage

The final conceptual model of the data processing process is shown in Figure 3.

6. Functions of the data processing system

The main goal of the system is to automate the calculation of data. To achieve this goal, you must perform the following functions:

1. Data aggregation. The P-system should implement a mechanism for collecting user data from several sources into one central one.
2. Data storage. The collected data of the P-system should be kept for later use.
3. Uploading data. The P-system should have a mechanism for obtaining the final set of collected data for performing subsequent analysis and calculating new data.
4. Calculation of new data. The P-system must implement the required processing and calculation algorithm to obtain a new data set in line with user requirements..
5. Saving the calculation results. The P-system should save the calculated results.

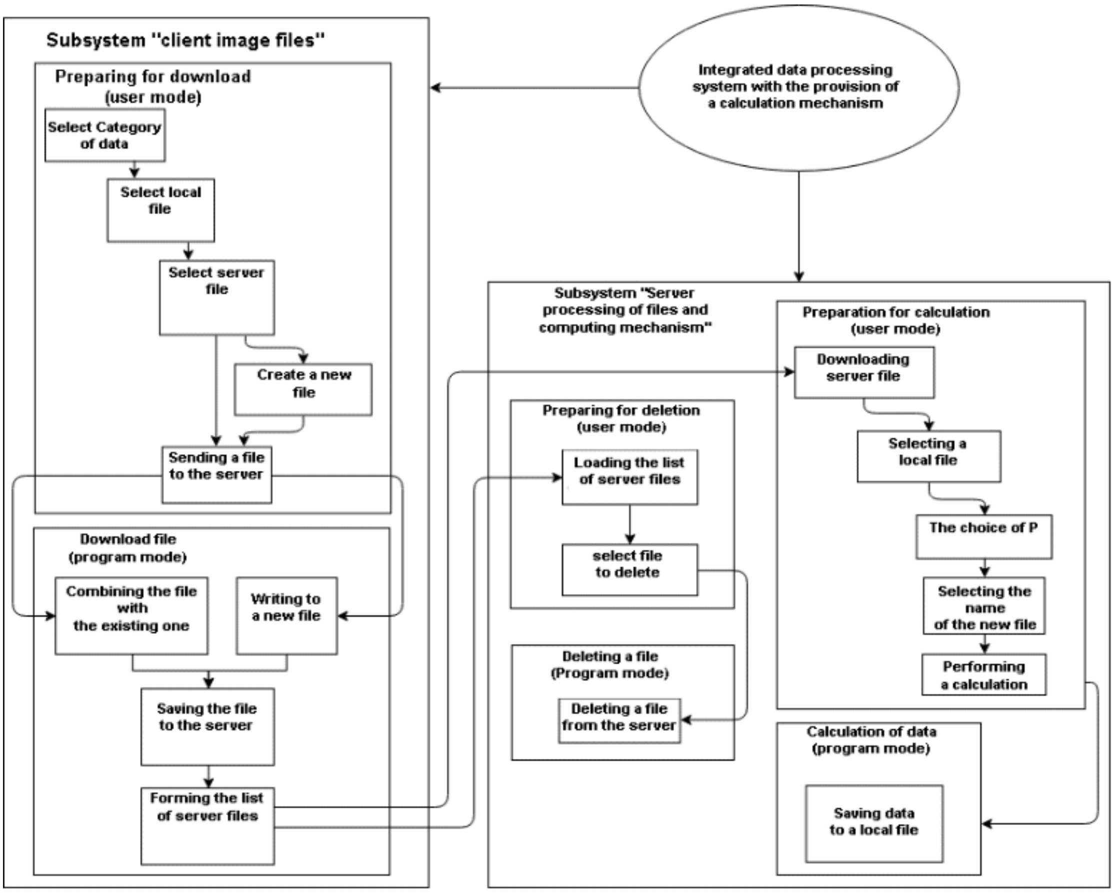
Subsystem "client image files"
Preparing for download (user mode)
Select Category of data
Select local file
Select server file
Create a new file
Sending a file to the server
Download file (program mode)
Combining the file with the existing one
Writing to a new file
Saving the file to the server
Forming the list of server files
Integrated data processing system with the provision of a calculation mechanism
Subsystem "Server processing of files and computing mechanism"
Preparation for calculation (user mode)
Downloading server file
Selecting a local file
The choice of P
Selecting the name of the new file
Performing a calculation
Preparing for deletion (user mode)
Loading the list of server files
select file to delete
Deleting a file (Program mode)
Deleting a file from the server
Calculation of data (program mode)
Saving data to a local file

Fig. 2. Structural scheme of the data processing

Рис. 2. Структурная схема процесса обработки данныхц

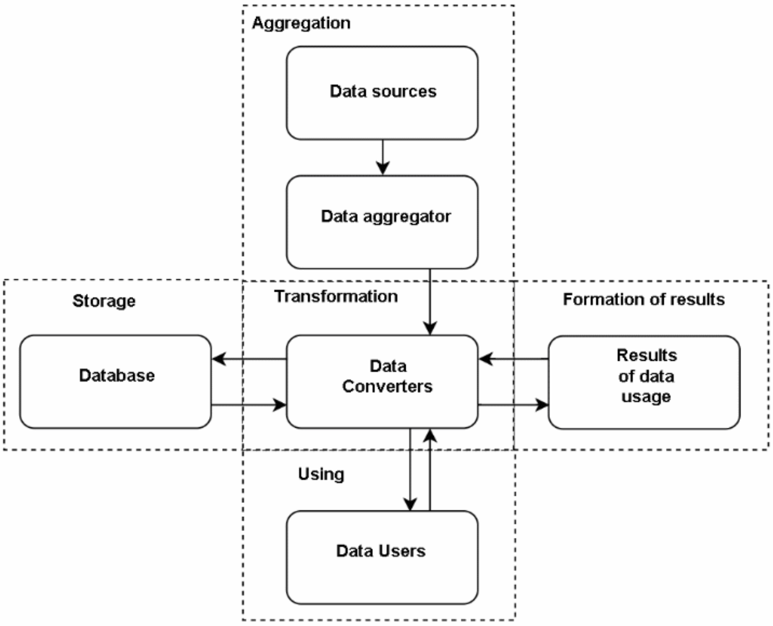

Fig. 3. Structural diagram of the conceptual model of the data processing process

Рис. 3. Структурная схема концептуальной модели процесса обработки данных

The functional scheme of the P-system implementing the data processing processes is shown in Figure 4.

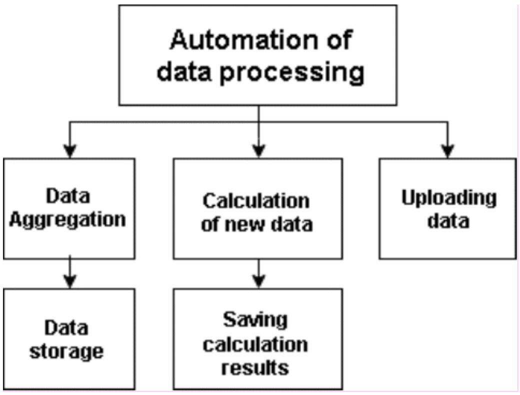

Fig. 4. Functional diagram of the data processing system
Рис. 4. Функциональная схема системы обработки данных

7. Basic functional modules of the IDPS

The P-system consists of the following basic modules:
1) Data access module (service)
2) The data interface module (head)
3) The interface module for calculating data (head)
4) File Combining Module (working)
5) Data verification module (manager)
6) Data calculation module (working)
7) Data storage module (service)
8) The module for deleting the incorrect data (working)
Let's consider in more detail each of the modules of the system and a list of descriptions of the tasks they solve.

8. Functions of IDPS modules

8.1. Data Access Module
1.1. creating a file on the server;
1.2. transferring a file from the server to the computer;
1.3. deleting the server file;
1.4. reading from the server file;
1.5. adding data to an existing server file.
8.2. Data federations interface module
2.1. demonstration of the main data about the file being sent;
2.2. demonstration of the field for entering a new name of the server file;
2.3. demonstration of the status of sending the file to the server;
2.4. demonstration of data about the region being deleted.
8.3. Calculation interface module
3.1. demonstration of the window for selecting two options for working with files;
3.2. demonstration of the field for selecting a server file and performing actions on it;
3.3. demonstration of the status of the settlement;
3.4. demonstration of the basic data for the calculation.
8.4. File Merge Module
4.1. merging of a local file of the first category with a server file of the first category;
4.2. merging of a local file of the second category with a server file of the second category.
8.5. Data Validation Module
5.1. checking the filename;
5.2. checking file extension;
5.3. checking the data category in the file;
5.4. checking the existence of the file.
8.6. Data calculation module
6.1. calculation according to the data of the first category;
6.2. calculation of the second category;
6.3. creating a file with the results of the calculation.
8.7. Storage Module
7.1. file storage;
7.2. files transfer.
8.8. The module for deleting incorrect data
8.1. removing incorrect data from the server file;
8.2. deleting the server file.

9. Mechanism for configuring and reconfiguring of the software system

To define the configuration mechanism, we define the categories of users. By the category of users we mean a large, usually not strictly outlined class, in its comparison with other similar classes [5]. In this system, we distinguish three categories of users:
- a user who has access to create and modify server files;
- a user with access to the removal / unloading of information from the server and making calculations;
- an administrator with full rights and access.
Depending on the user's category, the P-system must provide him with a number of possible operations, determined in a particular configuration of the system. We denote the configuration variants by the symbols Ψ1, Ψ2, Ψ3 (Figure 5). For users of the first category (configuration Ψ1), a mechanism for configuring and reconfiguring of work files is provided. For users of the second category (configuration Ψ2), a mechanism for configuring the calculated data is provided. For users of the third category (configuration Ψ3), mechanisms for configuring and reconfiguring of work files are provided, as well as a mechanism for configuring the calculated data.

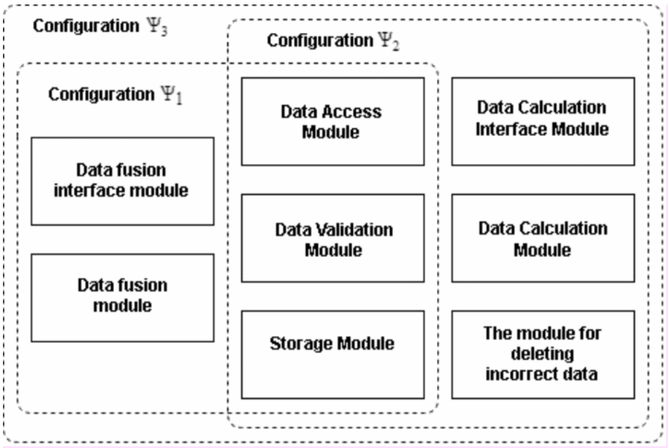

Fig. 5. Schematic of variants of software system configurations oriented to different categories of users
Рис. 5. Схема вариантов конфигураций программной системы, ориентированных на различные категории пользователей

1. Subsystem for client file processing (KO).

The KO subsystem includes the following modules:

1) data aggregation interface module;
2) data access module;
3) file merging module;
4) the data verification module;
5) data storage module.

The task of the subsystem is to allow the user to merge several files with a strict data organization structure and send the resulting file to the server. During the work, the files selected by the user for downloading must undergo all necessary checks (the correctness and category of the data contained in them, the file format, etc.). In case of unsuccessful verification, the file should not be sent to the server.

Thus, during the configuration and reconfiguration process, the user can create and edit work files and then send them to the server.

In the KO subsystem, the single configuration mode Ψ1 is implemented, in this mode a user can get the access to the server files and combine them with local ones.

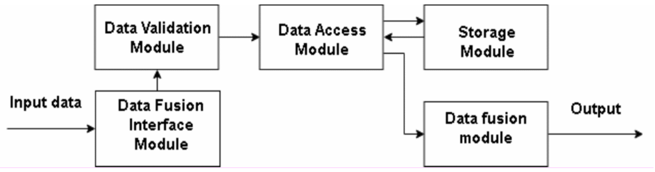

Fig. 6. Structural diagram of the configuration mode Ψ1

Рис. 6. Структурная схема режима работы конфигурации Ψ1

2. Subsystem of server processing of files with a computational mechanism (CO).

The list of modules that are part of the CO subsystem:

1) data interface interface module;
2) data access module;
3) module for calculating the data;
4) data storage module;
5) data verification module;
6) module for deleting the incorrect data.

The subsystem CO provides the user with the ability to perform complex mathematical calculations based on the data received from the server file. Calculations can be performed for two possible categories of data, so the subsystem provides a mechanism for selecting the necessary chain of calculations, depending on the type of data from the file received by the program.

Two modes of functioning of the Ψ2 configuration are implemented in the CO subsystem. The composition and interaction of the modules are shown in Figure 7 and Figure 8, respectively.

In mode I, the user can download server files and perform calculations with data received from local files. Mode II allows the user to remove incorrect data from server files.

The team-work of the two subsystems allows the user to access all the functionality developed within the framework of the considered IDPS. In this case, the configuration Ψ3 takes place. It implements four possible modes of operation. Since the modes I, II, III coincide with the operating modes of the configurations Ψ1 and Ψ2, we will discuss in more detail only the IV mode of operation of configurations Ψ3 (Figure 9.).

In the IV mode of the configuration Ψ3, the user has access to all possible operations: merge files, upload files to the server and download them from the server, perform calculations with data from local files, and delete incorrect data from server files.

In this system, the configuration mechanism was used as a means of adapting the software system to the tasks being solved. Due to the reorganization of the structure of the system through the formation of configurations from dedicated modules, it became possible to differentiate the availability of the system functionality for different categories of users without making changes to the program code.

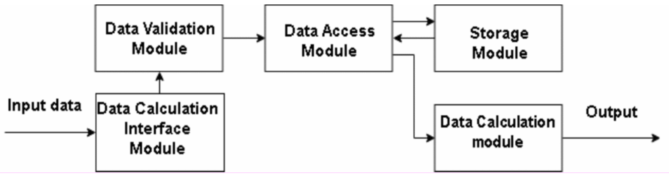

Fig. 7. Block diagram of the I mode of the configuration Ψ2 operation

Рис. 7. Структурная схема режима I работы конфигурации Ψ2

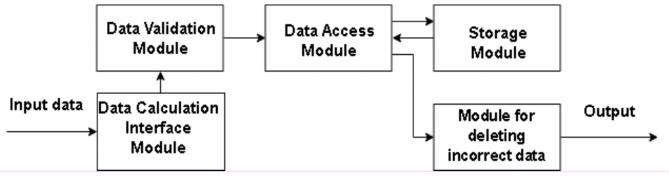

Fig. 8. Block diagram of the II mode of the configuration Ψ2 operation

Рис. 8. Структурная схема режима II работы конфигурации Ψ2

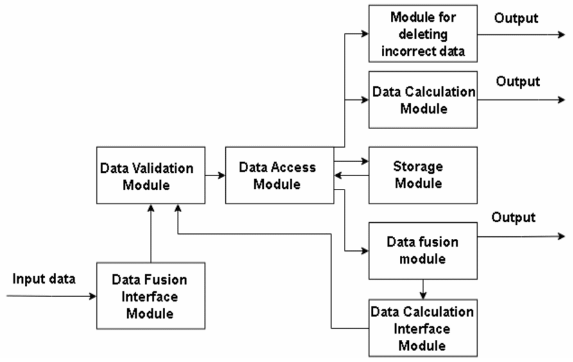

Fig. 9. Block diagram of the operation mode IV of the configuration Ψ3

Рис. 9. Структурная схема режима IV работы конфигурации Ψ3

Conclusion

The article considers the method of configuration modeling, focused on solving the problems of adaptation of modular software systems. Configuration modeling makes it possible to create software systems that adapt to the changing needs of users through an embedded - internal or independent external software component - the configurator. The adaptation process is implemented as a result of changing the operating modes of the system modules in accordance with the data flow received from the external environment. Thanks to the configurator, the P-system can automatically change its state to the target-oriented meeting the new requests of a particular user. Theoretical and methodical aspects of the use of the method of configuration modeling are tested on the example of a complex P-system for data processing. Positive results were achieved. It should be noted that the proposed approach to the creation of P-systems dramatically reduces the cost of their support and maintenance in the process of their operation.

Acknowledgements

The research is carried out with financial support of the Russian Foundation for Basic Research within the scientific project No. 16-29-04326 ofi-m.

References

[1] Leroy X. A modular module system. *Journal of Functional Programming*. 2000; 10(3):269-303. DOI: 10.1017/S0956796800003683

[2] Shibanov S.V., Mezenkov A.A., Shevchenko O.A., Ilyushkin A.S. Organizational and functional principles of active packets for information exchange and distributed application configuration. *University proceedings. Volga region. Technical sciences.* 2013; 1(25):5-18. Available at: https://elibrary.ru/item.asp?id=19033948 (accessed 01.05.2018). (In Russian)

[3] Rastrigin L.A. Adaptation of complex systems. Riga: Zinatne, 1981. 375 p. (In Russian)

[4] Bakanov A.B., Drozhdin V.V., Zinchenko R.E., Kuznetsov R N. Adaptation methods and generations of software development. *Izv. Penz. gos. pedagog. univ. im.i V. G. Belinskogo*. 2009; № 13(17):66-69. Available at: https://elibrary.ru/item.asp?id=13051113 (accessed 01.05.2018). (In Russian)

[5] Nechaev V.V. Configurational modeling: Part I. Theoretical aspects. Moscow: MIREA, 2007. 92 p. (In Russian)

[6] Vendrov A.M. Designing of the software of economic information systems. 2nd Edition. Moscow: Finance and Statistics, 2005. 544 p. (In Russian)

[7] Rogozov U.I., Sviridov A.S. The control in the structure of software application. *Informatization and communication.* 2012; 5:112-116. Available at: https://elibrary.ru/item.asp?id=18380683 (accessed 01.05.2018). (In Russian)

[8] Velichko Yu.I. The method of introducing the adaptation module into the user interface. *Problems of Information Technology*. 2014; 2(14):15-19. (In Russian)

[9] Lavrischeva K.M., Petrenko A.K. Software Product Lines Modeling. *Trudy ISP RAN/Proc. ISP RAS*. 2016; 28(6):49-64 (In Russian). DOI: 10.15514/ISPRAS2016-28(6)-4

[10] Valeev S.S., Ismagilova I.M. Statistical methods in the construction of adaptive interfaces in complex distributed technical systems. *CEUR Workshop Proceedings*. 2016; 1825:260-268. Available at: http://ceur-ws.org/Vol-1825/p33.pdf (accessed 01.05.2018). (In Russian)

[11] Kolesov Y.B., Senichenkov Yu.B. Modeling of systems. Object-oriented approach. St. Petersburg: BHV-Petersburg, 2006. 192 p. (In Russian)

[12] Kolesov Y.B., Senichenkov Yu.B. Modeling of systems. Dynamic and hybrid systems. St. Petersburg: BHV-Petersburg, 2006. 224 p. (In Russian)

[13] Dvoretskiy S.I., Muromtsev Yu.L., Pogonin V.A., Skhirtladze A.G. Modeling of systems. Moscow: Academy, 2009. 320 p. (In Russian)

[14] Algazinov E.K., Sirota A.A. Analysis and computer modeling of information processes and systems. Moscow: Dialog-MIFI, 2009. 416 p. (In Russian)

[15] Tyukin I.Yu., Terekhov V.A. Adaptation in nonlinear dynamical systems. Moscow: Izd-vo LKI, 2014. 384 p. (In Russian)

[16] Bellagio D., Milligan T. Software Development. Change management. Moscow: DMK Press, 2016. 384 p. (In Russian)

[17] Nechaev V.V., Koshkarev M.I. Smart problem solver: comparative analysis and architectural model. *Information and Telecommunication Technologies*. 2014; 21:51-61. (In Russian)

[18] Ramírez A., Romero J.R., Ventura S. Interactive multi-objective evolutionary optimization of software architectures. *Information Sciences*. 2018; 463–464:92-109. DOI:10.1016/j.ins.2018.06.034

[19] Brogi A., Canal C., Pimentel E. On the specification of software adaptation. *Electronic Notes in Theoretical Computer Science*. 2004; 97:47-65. DOI: 10.1016/j.entcs.2004.04.031

[20] Kuhrmann M., Ternité T., Friedrich J., Rausch A., Broy M. Flexible software process lines in practice: A metamodel-based approach to effectively construct and manage families of software process models. *Journal of Systems and Software*. 2016; 121:49-71. DOI: 10.1016/j.jss.2016.07.031

[21] Salama M., Bahsoon R. Analysing and modelling runtime architectural stability for self-adaptive software. *Journal of Systems and Software*. 2017; 133:95-112. DOI: 10.1016/j.jss.2017.07.041

[22] Hussein M., Nouacer R., Radermacher A. Safe adaptation of vehicle software systems. *Microprocessors and Microsystems*. 2017; 52:272-286. DOI: 10.1016/j.micpro.2017.06.014

[23] Bashari M., Bagheri E., Du W. Self-adaptation of service compositions through product line reconfiguration. *Journal of Systems and Software*. 2018; 144:84-105. DOI: 10.1016/j.jss.2018.05.069

[24] Bartusevics A., Novickis L. Models for Implementation of Software Configuration Management. *Procedia Computer Science*. 2015; 43:3-10. DOI: 10.1016/j.procs.2014.12.002

[25] Bajunaid N., Menascé D.A. Efficient modeling and optimizing of checkpointing in concurrent component-based software systems. *Journal of Systems and Software*. 2018; 139:1-13. DOI: 10.1016/j.jss.2018.01.032

[26] Behjati R., Nejati S. Architecture-level configuration of industrial control systems: Foundations for an efficient approach. *Science of Computer Programming*. 2018; 160:30-47. DOI: 10.1016/j.scico.2017.10.001

[27] Horcas J.-M., Pinto M., Fuent L. Variability models for generating efficient configurations of functional quality attributes. *Information and Software Technology*. 2018; 95:147-164. DOI: 10.1016/j.infsof.2017.10.018

Submitted 01.05.2018; revised 10.06.2018; published online 30.06.2018.

Список использованных источников

[1] *Leroy* X. A modular module system // Journal of Functional Programming. 2000. Vol. 10, issue 3. Pp. 269—303. DOI: 10.1017/S0956796800003683

[2] *Шибанов С.В., Мезенков А.А., Шевченко О.А., Илюшкин А.С.* Принципы организации и функционирования активных пакетов для обмена информацией и конфигурирования распределенных приложений // Известия высших учебных заведений. Поволжский регион. Технические науки. 2013. № 1(25). С. 5-18. URL: https://elibrary.ru/item.asp?id=19033948 (дата обращения: 01.05.2018).

[3] *Растригин Л.А.* Адаптация сложных систем. Рига: Зинатне, 1981. 375 с.

[4] *Баканов А.Б., Дрождин В.В., Зинченко Р.Е., Кузнецов Р.Н.* Методы адаптации и поколения развития программного обеспечения // Известия ПГПУ им. В. Г. Белинского. 2009. № 13(17). С. 66-69. URL: https://elibrary.ru/item.asp?id=13051113 (дата обращения: 01.05.2018).

[5] *Нечаев В.В.* Конфигурационное моделирование: Часть I. Теоретические аспекты. М.: МИРЭА, 2007. 92 с.

[6] *Вендров А.М.* Проектирование программного обеспечения экономических информационных систем. 2-е изд., перераб. и доп. М.: Финансы и статистика, 2005. 544 с.

[7] *Рогозов Ю.И., Свиридов А.С.* Элементы управления в структуре информационной системы // Информатизация и связь. 2012. № 5. С. 112-116. URL: https://elibrary.ru/item.asp?id=18380683 (дата обращения: 01.05.2018).

[8] *Величко Ю.И.* Метод внедрения модуля адаптации в пользовательский интерфейс // Проблеми інформаційних технологій. 2014. Т. 2, № 14. С. 15-19.

[9] *Лаврищева Е.М., Петренко А.К.* Моделирование семейств программных систем // Труды ИСП РАН. 2016. Том 28, вып. 6. С. 49-64. DOI: 10.15514/ISPRAS2016-28(6)-4

[10] *Валеев С.С., Исмагилова И.М.* Построение адаптивных интерфейсов в сложных распределенных технических системах с применением статистических методов // CEUR Workshop Proceedings. 2016. Vol. 1825. Pp. 260-268. URL: http://ceur-ws.org/Vol-1825/p33.pdf (дата обращения: 01.05.2018).

[11] *Колесов Ю.Б., Сениченков Ю.Б.* Моделирование систем. Объектно-ориентированный подход. СПб.: БХВ-Петербург, 2006. 192 с.

[12] *Колесов Ю.Б., Сениченков Ю.Б.* Моделирование систем. Динамические и гибридные системы. СПб.: БХВ-Петербург, 2006. 224 с.

[13] *Дворецкий С.И., Муромцев Ю.Л., Погонин В.А., Схиртладзе А.Г.* Моделирование систем. М.: Академия, 2009. 320 с.

[14] *Алгазинов Э.К., Сирота А.А.* Анализ и компьютерное моделирование информационных процессов и систем. М.: Диалог-МИФИ, 2009. 416 с.

[15] *Тюкин И.Ю., Терехов В.А.* Адаптация в нелинейных динамических системах. М.: Изд.-во ЛКИ, 2014. 384 с.

[16] *Белладжио Д., Миллиган Т.* Разработка программного обеспечения. Управление изменениями. М.: ДМК Пресс, 2016. 384 с.

[17] *Нечаев В.В., Кошкарёв М.И.* Интеллектуальные решатели задач: сравнительный анализ и архитектурная модель // Информационные и телекоммуникационные технологии. 2014. №21. С. 51-61.

[18] *Ramírez A., Romero J.R., Ventura S.* Interactive multi-objective evolutionary optimization of software architectures // Information Sciences. 2018. Vol. 463–464. Pp. 92-109. DOI:10.1016/j.ins.2018.06.034

[19] *Brogi A., Canal C., Pimentel E.* On the specification of software adaptation // Electronic Notes in Theoretical Computer Science. 2004. Vol. 97. Pp. 47-65. DOI: 10.1016/j.entcs.2004.04.031

[20] *Kuhrmann M., Ternité T., Friedrich J., Rausch A., Broy M.* Flexible software process lines in practice: A metamodel-based approach to effectively construct and manage families of software process models // Journal of Systems and Software. 2016. Vol. 121. Pp. 49-71. DOI: 10.1016/j.jss.2016.07.031

[21] *Salama M., Bahsoon R.* Analysing and modelling runtime architectural stability for self-adaptive software // Journal of Systems and Software. 2017. Vol. 133. Pp. 95-112. DOI: 10.1016/j.jss.2017.07.041

[22] *Hussein M., Nouacer R., Radermacher A.* Safe adaptation of vehicle software systems // Microprocessors and Microsystems. 2017. Vol. 52. Pp. 272-286. DOI: 10.1016/j.micpro.2017.06.014

[23] *Bashari M., Bagheri E., Du W.* Self-adaptation of service compositions through product line reconfiguration // Journal of Systems and Software. 2018. Vol. 144. Pp. 84-105. DOI: 10.1016/j.jss.2018.05.069

[24] *Bartusevics A., Novickis L.* Models for Implementation of Software Configuration Management // Procedia Computer Science. 2015. Vol. 43. Pp. 3-10. DOI: 10.1016/j.procs.2014.12.002

[25] *Bajunaid N., Menascé D.A.* Efficient modeling and optimizing of checkpointing in concurrent component-based software systems // Journal of Systems and Software. 2018. Vol. 139. Pp. 1-13. DOI: 10.1016/j.jss.2018.01.032

[26] *Behjati R., Nejati S.* Architecture-level configuration of industrial control systems: Foundations for an efficient approach // Science of Computer Programming. 2018. Vol. 160. Pp. 30-47. DOI: 10.1016/j.scico.2017.10.001

[27] *Horcas J.-M., Pinto M., Fuent L.* Variability models for generating efficient configurations of functional quality attributes // Information and Software Technology. 2018. Vol. 95. Pp. 147-164. DOI: 10.1016/j.infsof.2017.10.018

Поступила 01.05.2018; принята в печать 10.06.2018;
опубликована онлайн 30.06.2018.

Об авторах:

Нечаев Валентин Викторович, кандидат технических наук, профессор, профессор кафедры инструментального и прикладного программного обеспечения, МИРЭА – Российский технологический университет (119454, Россия, г. Москва, пр. Вернадского, д. 78), ORCID: http://orcid.org/0000-0001-7171-3874, nechaev@mirea.ru

Баширов Антон Станиславович, студент кафедры инструментального и прикладного программного обеспечения, МИРЭА – Российский технологический университет (119454, Россия, г. Москва, пр. Вернадского, д. 78), ORCID: http://orcid.org/0000-0002-4113-0704, antonbashir@mail.ru

Лебедева Наталия Игоревна, студент кафедры инструментального и прикладного программного обеспечения, МИРЭА – Российский технологический университет (119454, Россия, г. Москва, пр. Вернадского, д. 78), ORCID: http://orcid.org/0000-0003-3421-4923, natalia.i.lebedeva@yandex.ru

Федин Михаил Андреевич, студент кафедры инструментального и прикладного программного обеспечения, МИРЭА – Российский технологический университет (119454, Россия, г. Москва, пр. Вернадского, д. 78), ORCID: http://orcid.org/0000-0001-9398-8297, vbif.32@ya.ru