**Адаптация программных систем к задачам пользователей на основе**

**метода конфигурационного моделирования**

***В.В. Нечаев, А.С. Баширов, Н.И. Лебедева, М.А. Федин***

**Аннотация**

В работе рассматривается задача адаптации программных систем (ПС) и комплексов (ПК) на основе метода конфигурационного моделирования (МКМ). Проводится краткое описание МКМ и его применения. Описывается концептуальная модель ПС и осуществляется её интерпретация в форме комплексной системы обработки данных (КСОД). На примере КСОД реализуется решение задачи адаптации ПС с элементами динамического конфигурирования по входным данным.

**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. A brief description of MKM and its application is given. The conceptual model of the PS is described and its interpretation is implemented in the form of an integrated data processing system (CSD). On the example of the CSD, the solution of the problem of adaptation of the PS with dynamic configuration elements by input data is realized.

**Ключевые слова**

Адаптация, конфигурационное моделирование, концептуальное моделирование, моделирование систем, программа, конфигурация

**Keywords**

Adaptation, configuration modeling, conceptual modeling, system modeling, program, configuration

**Введение**

Современные большие программные системы (П-системы) разрабатываются и поддерживаются в условиях постоянно развивающихся требований. Внесение изменений в существующие П- системы всегда связано с дополнительными затратами, возрастающими с течением времени. Именно стремление сократить объём таких изменений стало основной причиной создания адаптивных программных (АП) систем. АП-системы должны обладать свойством целенаправленного развития под воздействием внешних условий. При этом цель функционирования АП-системы была достигнута, несмотря на изменения. Для обеспечения выполнения таких требований наиболее адекватна модульная архитектура АП-системы. Она даёт возможность легко обмениваться данными с обслуживаемой системой [1]. Адаптация выступает в качестве средства управления объектом при отсутствии его точной модели. Адаптация может осуществляться посредством использования методов: параметрической, организационной, структурной адаптации, а также эволюционных алгоритмов [2,3]. Именно адаптивным программным системам и их практической реализации посвящена данная статья.

В работе для решения задачи адаптации программной системы авторы используют метод конфигурационного моделирования [4]. Конфигурация – это качественная характеристика структуры, определяющая ее пространственную, логическую и/или временную, а также комбинированную (комплексную) организацию П-системы. С концептуальной точки зрения конфигурация – это структура структур или метаструктура. Программную систему можно определить, как конфигурируемую, если её можно настроить без программирования дополнительных функций и/или без изменения исходного кода программы. Конфигурационное моделирование, как метод адаптации модульных программных систем, даёт возможность существенного расширения их функционала и снижения дополнительных расходов при разработке программных продуктов. Наличие в АП-системе механизмов конфигурирования обеспечивает гибкость использования программного обеспечения в рамках поставленных задач и существующей среды. Отметим, что метод конфигурирования изначально использовался для формирования конфигураций вычислительных систем, комплексов и сетей [2,4]. Однако он легко может быть использован и применительно к программным системам. Возможность конфигурирования программных продуктов существует достаточно давно. Однако создание П-систем, в структуру которых изначально заложены широкие возможности для адаптации при наличии определённых целей, стало использоваться сравнительно недавно.

1. **Цель работы**

*Цель работы состоит в рассмотрении особенностей метода конфигурационного моделирования применительно к гибким программным системам, использования этого метода в качестве инструмента адаптации таких П-систем к решаемым пользователем задачам, а также демонстрации применимости механизма конфигурирования на примере создания собственной программной системы*

Работа включает в себя: анализ актуальности вопроса создания демонстрируемой системы; описание адаптации программных систем на основе конфигурационного моделирования; концептуальную модель адаптивной программной системы; реализацию АП-системы на примере комплексной системы обработки данных (КСОД).

1. **Актуальность разработки адаптивных программных систем.**

Современные программные системы широко используются практически во всех сферах жизнедеятельности общества. Потребности пользователей возрастают постоянно. Функционал, который выполняла П-система вчера, сегодня может оказаться уже недостаточным или не актуальным. Таким образом, возникает необходимость обновления уже введенной в эксплуатацию П-системы. Отметим, объёмы исходных кодов современных П-систем достигают размеров, при которых при суммировании всего программного кода среднестатистической П-системы в одном файле, её невозможно поддерживать. В итоге, такая ситуация приводит к возрастанию сложности распределения функционала П-системы в соответствии с потребностями пользователей. [5]

1. **Метод конфигурационного моделирования в**

**задачах адаптации П-систем**

Для решения поставленной задачи в данной работе используется метод структурной адаптации. П-система реализуется на основе модульного подхода. Для адаптации используется метод конфигурационного моделирования., С целью конкретизации, в качестве примера, определяются две задачи адаптации. Первая - это задача обновления существующей активной программной системы. Вторая - задача декомпозиция активного функционала программной системы соответствии с потребностями пользователя. Адаптация достигается посредством исполнения механизма конфигурирования [6]. Механизм конфигурирования заключается в реформировании структуры программной системы посредством изменения текущего набора её модулей и их взаимосвязей. Управление конфигурациями П-системы обеспечивается через отдельный программный модуль – *конфигуратор.* Конфигуратор (**К**) – это компонент П-системы, назначение которого – конфигурирование и реконфигурирование основной части П-системы, при этом предполагается что П-система может находиться в различных состояниях и имеет несколько различных режимов работы. Под режимом работы понимается совокупность текущих параметров П-системы, определяющих её состояние, а также наборы задач, которые система решает, находясь в том или ином режиме.

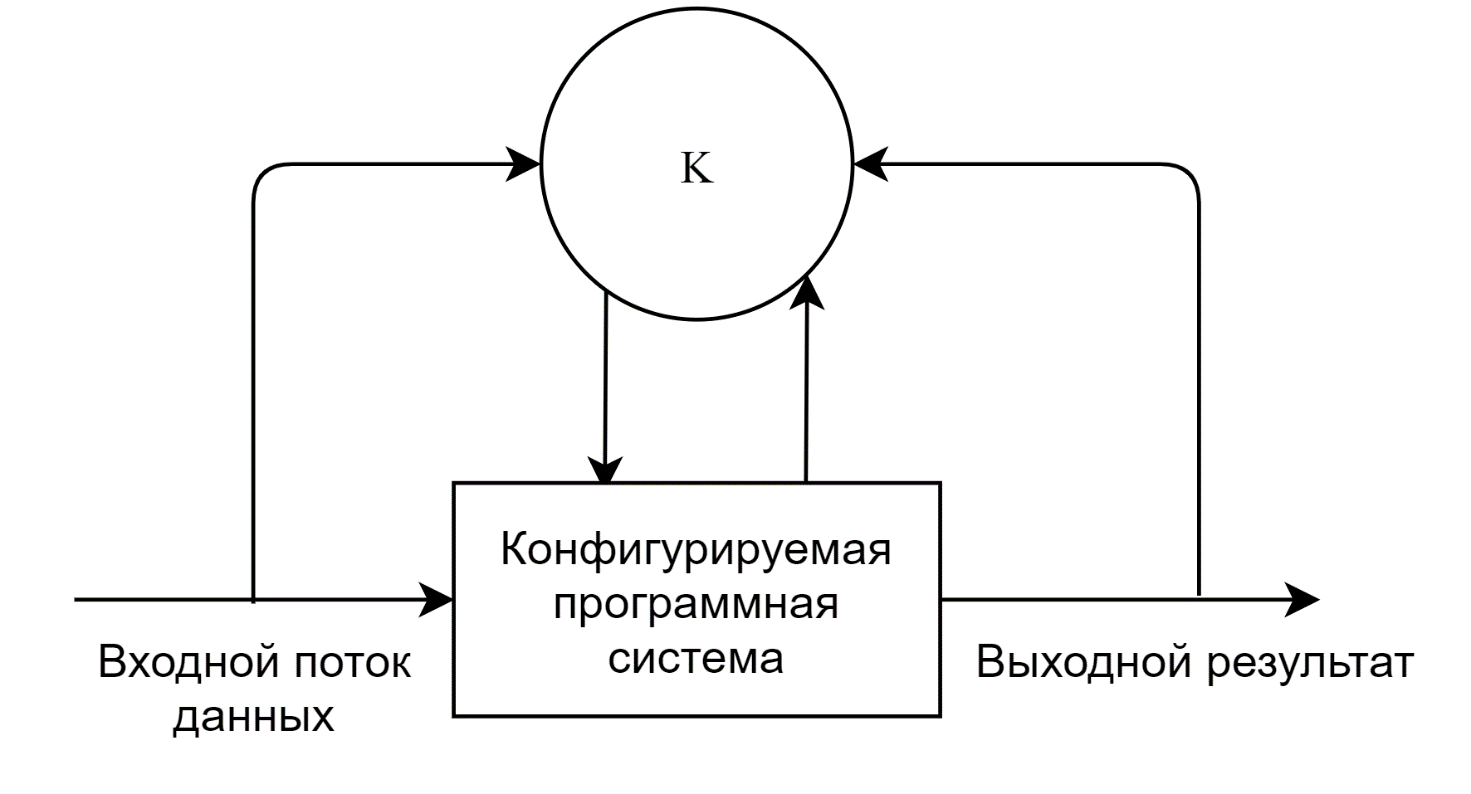
В роли конфигуратора **К** также может выступать и внешняя по отношения к П-системе программа, работающая в той же среде, что и конфигурируемая. Однако, следует учитывать, что при наличии конфигуратора **К**, как отдельного от П-системы программного компонента, возникает задача создания канала управления, который обеспечивал бы без потерь передачу управляющей информации от конфигуратора **К** к конфигурируемой П-системе, а также осведомительной информации (потребности) от П-системы к конфигуратору **К** (рис 1.). Использование модульной архитектуры П-системы в сочетании с конфигуратором **К** позволяет выполнять частичные обновления П-системы в рамках ограниченного количества модулей, не затрагивая остальные части системы. Тем самым решается поставленная выше задача адаптации П-системы по обновлению активного её экземпляра [7].

Рис. 1. Схема П-системы управления конфигурациями.

Задача декомпозиции функционала П-системы в соответствии с потребностями пользователя решается посредством использование конфигурируемых режимов её работы. Другими словами, используя конфигуратор **К** мы можем изменить текущую конфигурацию модулей в соответствии с потребностями пользователя в данный момент времени или же, из соображений безопасности, ограничив доступную пользователю область функционала системы выделенным набором модулей. В условиях изменяющихся требований, предъявляемых к системе, достоинствами использования такого подхода являются:

– отсутствие необходимости внесения доработок в уже существующую систему, что может привести к увеличению стоимости проекта и сроков его внедрения;

– возможность внесения изменений в функционал уже существующей системы посредством выделения режимов её работы, определяемых структурой и составом взаимодействующих модулей без изменений самих модулей;

– отсутствие необходимости создания системы «с нуля» для удовлетворения новых требований.

В качестве примера практической реализации задачи адаптации посредством конфигурационного моделирования рассмотрим *комплексную систему обработки данных* (КСОД).

1. **Комплексная система обработки данных**

Комплексная система обработки данных с предоставлением механизма расчёта состоит из двух подсистем: клиентской обработки файлов (КО) и серверной обработки файлов с вычислительным механизмом (СО).

Достоинство создания и использования конфигурируемой КСОД определяется тем, что она обладает рядом факторов, среди которых выделим следующие:

– возможность поэтапного ввода в эксплуатацию отдельного функционала, реализующего определенный круг задач автоматизации бизнес-процессов;

– автоматизация длительного процесса расчёта данных;

– уменьшение вероятности допущения ошибок, связанных с человеческим фактором при произведении расчётов;

– экономия ресурсов за счёт проектирования рациональной конфигурации системы.

Структурная схема процесса обработки данных представлена на рис. 2

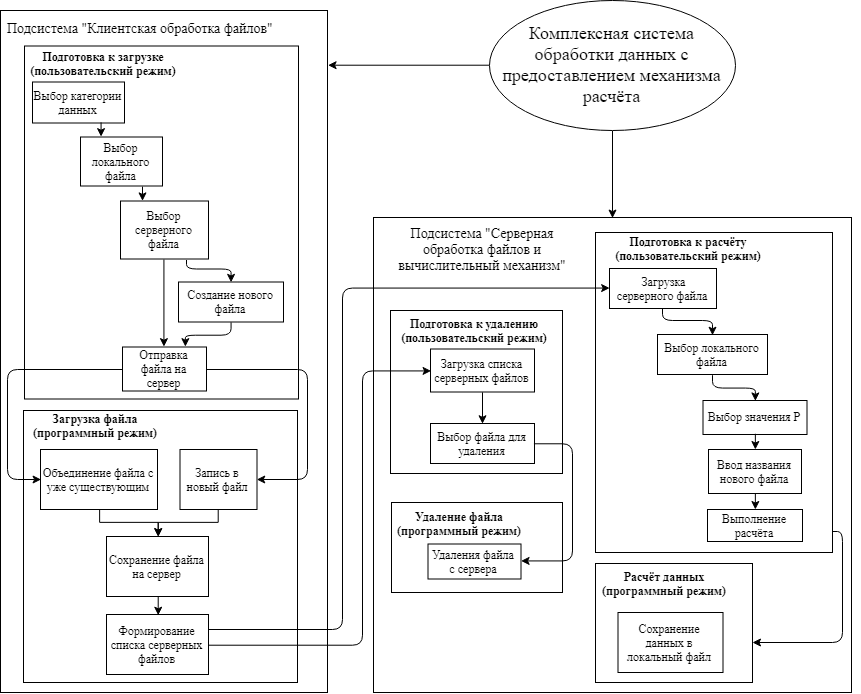


Рис.2. Структурная схема процесса обработки данных

1. **Структурная схема концептуальной модели**

**системы обработки данных**

Концептуальная модель системы в узком смысле представляет совокупность концепт компонентов системы и их взаимодействия.

Процесс обработки данных можно рассмотреть двояко:

1. как процесс, состоящий из нескольких действий;
2. программа как совокупность компонентов, где под компонентом понимается программный модуль-исполнитель действия по обработке данных.

С точки зрения процессного подхода, обработка данных состоит из следующего набора действий: агрегация данных, конвертация данных, хранение данных, использование данных, формирование результатов использования данных.

Процесс обработки данных может повторяться до тех пор, пока конечный результат не будет удовлетворять потребностям пользователя. При этом, каждая последующая итерация процесса будет применяться к результату действий использования или конвертации данных предыдущего Адве результата.

Компонентная схема процесса обработки данных выглядит следующим образом:

где

* – множество источников данных,
* – агрегатор данных
* – множество преобразователей данных
* – база данных
* – множество пользователей данных
* – множество результатов использования данных

Итоговая концептуальная модель процесса обработки данных представлена на рис 3.

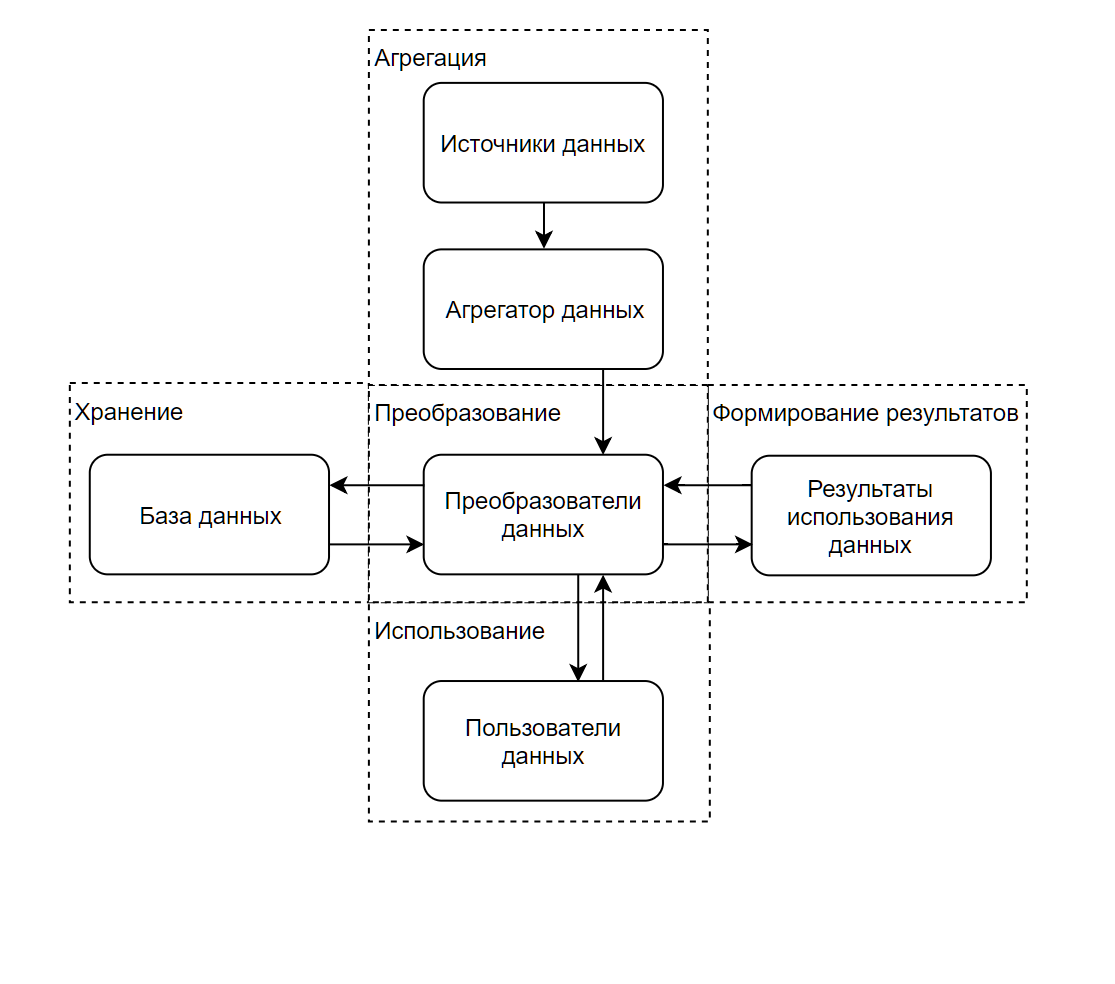


Рис 3. Структурная схема концептуальной модели

процесса обработки данных

1. **Функции системы обработки данных**

Основная цель системы заключается в автоматизации процесса расчёта данных. Для реализации этой цели необходимо выполнить следующие функции:

1. Агрегация данных. ПС должна реализовывать механизм сбора данных пользователей из нескольких источников в один центральный.
2. Хранение данных. Собранные данные ПС должна сохранить для последующего применения.
3. Выгрузка данных. ПС должна иметь механизм получения конечного набора собранных данных для выполнения последующего анализа и расчёта новых данных.
4. Расчёт новых данных. ПС должна реализовывать требуемый алгоритм обработки и расчёта для получения нового набора данных в соответствие с потребностями пользователя.
5. Сохранение результатов расчёта. ПС должна сохранить полученные результаты расчёта.

Функциональная схема ПС, реализующей процессы обработки данных представлены на рис 4.

****

Рис 4. Функциональная схема системы обработки данных

1. **Базовые функциональные модули КСОД**

Рассматриваемая ПС состоит из следующих базовых модулей:

1) Модуль доступа к данным (сервисный)

2) Модуль интерфейса объединения данных (головной)

3) Модуль интерфейса расчёта данных (головной)

4) Модуль объединения файлов (рабочий)

5) Модуль проверки данных (управляющий)

6) Модуль расчёта данных (рабочий)

7) Модуль хранения данных (сервисный)

8) Модуль удаления некорректных данных (рабочий)

Рассмотрим более подробно каждый из модулей системы и приведем описание решаемых ими задач.

1. **Функции модулей КСОД**
   1. **Модуль доступа к данным**

1.1. создание файла на сервере;

1.2. передача файла с сервера на компьютер;

1.3. удаление серверного файла;

1.4. чтение из серверного файла;

1.5. добавление данных в существующий серверный файл.

* 1. **Модуль интерфейса объединения данных**

2.1. демонстрация основных данных об отправляемом файле;

2.2. демонстрация поля для ввода нового имени серверного файла;

2.3. демонстрация статуса отправки файла на сервер;

2.4. демонстрация данных об удаляемом регионе.

* 1. **Модуль интерфейса расчёта данных**

3.1. демонстрация окна выбора двух опций работы с файлами;

3.2. демонстрация поля для выбора серверного файла и выполнения действий над ним;

3.3. демонстрация статуса выполнения расчёта;

3.4. демонстрация основных данных для выполнения расчёта.

* 1. **Модуль объединения файлов**

4.1. слияние локального файла первой категории с серверным файлом первой категории;

4.2. слияние локального файла второй категории с серверным файлом второй категории.

* 1. **Модуль проверки данных**

5.1. проверка имени файла;

5.2. проверка расширения файла;

5.3. проверка категории данных в файле;

5.4. проверка существования файла.

* 1. **Модуль расчёта данных**

6.1. выполнение расчёта по данным первой категории;

6.2. выполнение расчёта по данным второй категории;

6.3. создание файла с результатами расчёта.

* 1. **Модуль хранения данных**

7.1. хранение файлов;

7.2. передача файлов.

* 1. **Модуль удаления некорректных данных**

8.1. удаление некорректных данных из серверного файла;

8.2. удаление серверного файла.

1. **Механизм конфигурирования и**

**реконфигурирования программной системы**

Далее необходимо выделить категории пользователей. Под категорией пользователей будем понимать крупный, обычно не строго очерченный класс, в его сравнении с другими такими же классами [8]. В рассматриваемой системе выделим три категории пользователей:

- пользователь, обладающий доступом к созданию и изменению серверных файлов;

- пользователь с доступом к удалению/выгрузке информации с сервера и произведению расчётов;

- администратор, обладающий полными правами и доступами.

В зависимости от категории пользователя, ПС должна предоставлять ему ряд возможных операций, определяемых в той или иной конфигурации системы. Варианты конфигураций – Ѱ1, Ѱ2, Ѱ3 (рис.5). Для пользователей первой категории (вариант конфигурация Ѱ1) предоставляется механизм конфигурирования и реконфигурирования рабочих файлов. Для пользователей второй категории (вариант конфигурации Ѱ2) предоставляется механизм конфигурирования расчётных данных. Для пользователей третьей категории (вариант конфигурации Ѱ3) предоставляются механизмы конфигурирования и реконфигурирования рабочих файлов, а также механизм конфигурирования расчётных данных.

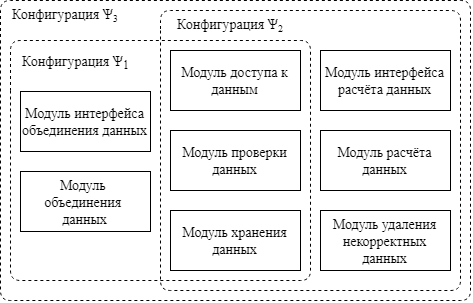


Рис. 5. Схема вариантов конфигураций программной системы,

ориентированных на различные категории пользователей

1. Подсистема клиентской обработки файлов

Подсистема (КО) включает в себя следующие модули:

1) модуль интерфейса объединения данных;

2) модуль доступа к данным;

3) модуль объединения файлов;

4) модуль проверки данных;

5) модуль хранения данных.

Задача подсистемы – предоставление пользователю возможности слияния нескольких файлов со строгой структурой организации данных и отправки результирующего файла на сервер. В ходе работы, выбранные пользователем файлы для загрузки должны пройти все необходимые проверки (корректность и категория содержащихся в них данных, формат файла и др.). В случае неудачной проверки, файл не должен отправляться на сервер.

Таким образом, в процессе конфигурирования и реконфигурирования пользователь имеет возможность создания и редактирования рабочих файлов с дальнейшей отправкой их на сервер.

В подсистеме КО реализован единственный режим конфигурации Ѱ1, в котором пользователь может получать доступ к серверным файлам и объединять их с локальными. Взаимодействие модулей в данном режиме представлено на рис.6.

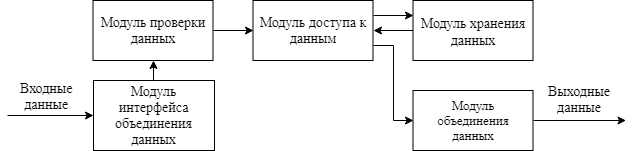


Рис. 6. Структурная схема режима работы конфигурации Ѱ1

1. Подсистема серверной обработки файлов с вычислительным механизмом (СО).

Список модулей, входящих в состав подсистемы СО:

1) Модуль интерфейса расчёта данных;

2) модуль доступа к данным;

3) модуль расчёта данных;

4) модуль хранения данных;

5) модуль проверки данных;

6) модуль удаления некорректных данных.

Подсистема СО предоставляет пользователю возможность выполнения сложных математических расчётов на основе данных, полученных из серверного файла. Расчёты могут выполняться для двух возможных категорий данных, поэтому в подсистеме предусмотрен механизм выбора необходимой цепочки вычислений, в зависимости от типа данных из полученного программой файла.

В подсистеме СО реализовано два режима функционирования конфигурации Ѱ2. Состав и взаимодействие модулей представлены на рис.7 и рис.8 соответственно.



Рис.7. Структурная схема режима I работы конфигурации Ѱ2



Рис.8. Структурная схема режима II работы конфигурации Ѱ2

В режиме I пользователь может скачивать серверные файлы и производить расчёты над данными, полученными из локальных файлов. Режим II позволяет пользователю производить операции удаления некорректных данных из серверных файлов.

Совместная работа двух подсистем позволяет пользователю получить доступ ко всему функционалу, разработанному в рамках рассматриваемой КСОД. В данном случае имеет место конфигурация Ѱ3. Она реализует четыре возможных режима работы. Так как режимы I, II, III совпадают с режимами работы конфигураций Ѱ1 иѰ2, рассмотрим более подробно лишь IV режим работы конфигураций Ѱ3 (рис.9.).

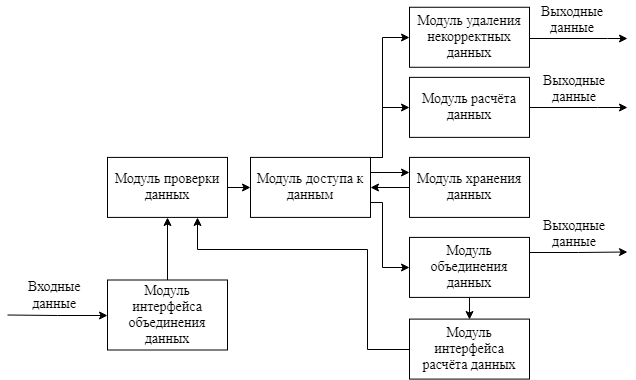


Рис. 9. – Структурная схема режима IV работы конфигурации Ѱ3

В режиме IV конфигурации Ѱ3 пользователь имеет доступ к воспроизведению всех возможных операций: объединять файлы, загружать файлы на сервер и скачивать их с сервера, производить расчёты над данными из локальных файлов и удалять некорректные данные из серверных файлов.

В рассматриваемой системе механизм конфигурирования использован как средство адаптации программной системы к решаемым задачам. Благодаря реорганизации структуры системы посредством формирования конфигураций из выделенных модулей, стало возможно разграничение доступности функционала системы для различных категорий пользователей без внесения изменений в программный код.

**Выводы**

В работе рассмотрено применение механизмов конфигурационного моделирования как одного из методов решения задачи адаптации программных систем. Конфигурационное моделирование даёт возможность создавать программные системы, адаптируемые к условиям функционирования посредством внутреннего или внешнего компонента – конфигуратора. Предложенный подход предполагает наличие в программной системе механизма адаптации под внутренние и внешние изменения посредством изменения режимов работы системных модулей в зависимости от данных, полученных из внешней среды. Благодаря конфигуратору, система способна автоматически изменять своё состояние на целевое, удовлетворяющее новым требованиям. Такой подход резко сокращает расходы на поддержку и обслуживание программных систем в процессе их эксплуатации.

**Список литературы**

1. С. В. Шибанов, А. А. Мезенков, О. А. Шевченко, А. С. Илюшкин - 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.
2. Нечаев В. В. Концептуальное модельное представление задачи как системы. Информационные технологии. – 2009, № 9 (157). - с. 26-32.
3. Л. А. Растригин. 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.
5. Вендров A.M. . Проектирование программного обеспечения экономических информационных систем: Учебник. — 2-е изд., перераб. и доп. - М.: Финансы и статистика, 2005. - 544 с.
6. Рогозов Ю.И., Свиридов А.С. Элементы управления в структуре информационной системы // Информатизация и связь. – 2012. – № 5. – С. 112-116.
7. Xavier Leroy. A Modular Module System // vol.10, issue 3. — Journal of Functional Programming, 2000. — С. 269—303.
8. В.В. Конфигурационное моделирование: часть I. Теоретические аспекты: Учебное пособие / Государственное образовательное учреждение высшего профессионального образования «Московский государственный институт радиотехники, электроники и автоматики (технический университет)». – М.: 2007. – 92 с.