**Титульник**

Содержание

[Введение 3](#_Toc510475281)

[1. Адаптация сложных программных систем к конкретным пользовательским задачам 4](#_Toc510475282)

[2. Методы адаптации программного обеспечения 5](#_Toc510475283)

[3. Адаптация через конфигурацию 6](#_Toc510475284)

[4. Модульность архитектуры программной системы 7](#_Toc510475285)

[5. Применение конфигурационного механизма для адаптации сложной системы. 9](#_Toc510475286)

[6. Конфигурационное моделирование 11](#_Toc510475287)

[7. Классификация задач конфигурационного моделирования 14](#_Toc510475288)

[8. Управление конфигурациями. Цели, задачи и основные действия 15](#_Toc510475289)

[Вывод 21](#_Toc510475290)

# Введение

В настоящее время пользователи заинтересованы в гибкости и масштабируемости программных систем, обеспечивающих работу в условиях постоянно изменяющееся внешней среды. Способность системы изменяться при изменении внешних условий так, чтобы цель её функционирования была достигнута, даже если изменения внешних условий препятствуют этому, называют адаптивностью.

Таким образом, можно сказать, что оптимальное функционирование системы при изменяемом внешнем окружении достигается благодаря процессам адаптации, конфигурирования и реконфигурирования. Конфигурирование и адаптация программных приложений необходимы для приведения их в соответствии с изменившимися требованиями [1, с 6]. При наличии в системе подобных механизмов, пользователь имеет возможность работать в нескольких различных режимах, что обеспечивает гибкость использования программного обеспечения в рамках существующей среды и поставленных задач.

Цель данной научной работы состоит в рассмотрении особенностей конфигурирования гибких адаптивных программных систем.

Работа включает в себя:

* Анализ актуальности вопроса создания адаптируемой системы
* Освещение методов адаптации программных систем
* Описание применения конфигурационного моделирования как одного из методов адаптации программных систем к решаемым задачам
* Рассмотрение основных целей и элементов управления конфигурациями

# 1. Адаптация сложных программных систем к конкретным пользовательским задачам

Сложные программные системы отличаются своей динамикой и непостоянством. Их внутренняя и внешняя среды способны к постоянному обновлению и изменению. В нестабильное состояние программу приводят как внешние факторы, такие как постоянные запросы на обработку информации, помехи в каналах связи ЭВМ, так и внутренние свойства: ненадёжность отдельных элементов и всей системы в целом. Подобное поведение порождает необходимость в реорганизации системы и её адаптации к решаемым задачам.

Задача адаптации программных систем сводится к построению грамотного процесса управления программой, позволяющего в зависимости от изменений среды корректировать её состояние, опираясь на заданные критерии.

Адаптация является одной из ключевых и сложных задач моделирования и проектирования программных систем, и представляет собой многоуровневый и поэтапный процесс. Выделяют пять основных уровней адаптации:

1. Аппаратный уровень. На данном уровне регулируются аппаратные параметры и аппаратная структура программы.

2. Алгоритмический уровень. Здесь объектом адаптации является алгоритмическая составляющая программы. Ключевая задача управления на этом уровне – реорганизация используемых в системе алгоритмов под новые требования решаемой задачи.

3. Программный уровень адаптации обеспечивает регулирование используемого множества доступных программ под требования к системе в данный момент времени.

4. Системный уровень адаптации улучшает функционирование обеспечения программного комплекса.

5. Сетевой уровень адаптации связан с адаптацией процессов передачи данных между программными компонентами.

Таким образом, можно сделать вывод, что адаптация к внешним условиям и внутренним состояниям программных систем является одной из ключевых задач их моделирования и проектирования, а, значит, именно от адаптации будет зависеть адекватность функционирования программы при любых внешних и внутренних изменениях [2, с.47].

# 2. Методы адаптации программного обеспечения

Выделим пять основных методов адаптации программного обеспечения [3, с. 68]:

1. Параметрическая адаптация подразумевает настройку входных параметров программного обеспечения, определяющих поведение и функционал программы. Необходимость в такого рода адаптации возникает ввиду изменения характеристик управляемого объекта.

2. Функциональная адаптация – изменение функций программного обеспечения. При функциональной адаптации возможна настройка параметров системы, при условии сохранения структуры и организации системы для выполнения задач, изначально заложенных в неё.

3. Структурная адаптация предполагает модификацию или замену одних структурных компонентов (модулей) системы на другие. При этом возможно использование как функциональной, так и параметрической адаптации системы.

4. Размножение позволяет системе порождать потомков со свойствами подобными родительским, но обладающими более широкими возможностями, что позволяет им эффективнее адаптироваться к существенным изменениям.

5. Эволюционная адаптация моделирует процесс биологической эволюции для программных систем, включающий 4 этапа: зарождение системы, становление системы определенного качества, устойчивое функционирование, деградацию и гибель системы.

Наиболее близкой к конфигурированию является структурная адаптация, которая позволяет программе становиться более адекватной решаемым пользователем задачам путём реорганизации её структурных компонентов.

# 3. Адаптация через конфигурирование

Конфигурирование и адаптацию программного обеспечения по смыслу часто путают. Программное обеспечение можно определить, как конфигурируемое, если его можно настроить без программирования дополнительных функций и/или без изменения исходного кода программы. Если требования к системе нельзя удовлетворить без программирования или изменения исходного кода, то программное обеспечение принято называть адаптируемым [4, с 37].

Конфигуратор – подсистема, ключевая цель которой – конфигурирование и реконфигурирование основной системы путём изменения режимов её работы. Под режимом работы понимается совокупность текущего состояния системы и набора задач, которые система должна решать, находясь в этом режиме.

Конфигуратор, в свою очередь, может быть реализован в рамках конфигурационного модуля. В этом случае саму систему можно считать автономной с точки зрения конфигурирования. Также в роли конфигуратора может выступать другая программная система, работающая в той же среде, что и основная. Однако, следует учесть, что при наличии отдельной системы-конфигуратора возникает задача создания канала управления, который позволял бы без потерь передавать указания конфигуратора к конфигурируемой системе, а также сигналы от системы к конфигуратору.

Базовым объектом, на который воздействует конфигуратор является программный модуль. Модуль – самостоятельная часть программы, имеющая определенное назначение и обеспечивающая заданные функции обработки автономно от других программных модулей [5]. Каждый модуль используется для достижения одной или нескольких целей системы, которые в свою очередь подразделяются на задачи. При создании программных продуктов выделяются многократно используемые модули, проводится их типизация и унификация, за счёт чего сокращаются сроки и трудозатраты на разработку программного продукта в целом. Совокупность системных модулей предоставляет полный набор функций для достижения главной цели системы.

Совокупность модулей и подмодулей даёт возможность согласовать процессы, происходящие в сложной системе, между собой в масштабах крупных предприятий и совершенствовать эти процессы по мере их развития.

Определение оптимальной конфигурации модулей системы исходя из поступающих от заказчика требований может служить эффективным средством реализации различных пользовательских задач. В условиях меняющихся требований, предъявляемых к системе, преимуществами использования механизмов конфигурации и реконфигурации являются:

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

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

# 4. Модульность архитектуры программной системы

Совокупность текущего состояния системы и выполняемого ей функционала является конфигурацией. На основе исходных параметров, с которыми была запущена система, конфигуратор устанавливает определённый режим работы для текущей сессии, выбирая подходящие для решения требуемых задач компоненты.

Под термином «архитектура», в большинстве случаев, понимают «некие конструктивные решения, которые после их принятия с трудом поддаются изменению» [6]. Архитектура определяет главные компоненты системы и способы их взаимодействия. Также это выбор решений, которые интерпретируются как основополагающие и не подлежащие изменению в будущем.

Создание структуры программы и построение её архитектуры подразумевает собой декомпозицию системы на подсистемы (функциональные модули, сервисы, слои, подпрограммы) и организацию взаимодействия этих подсистем друг с другом и внешней средой.

Ниже перечислены свойства, присущие правильно спроектированной архитектуре [7]:

* Масштабируемость – возможность расширять систему и увеличивать её производительность и функциональность, за счёт добавления новых модулей;
* Ремонтопригодность подразумевает, что модули системы являются независимыми друг от друга и изменение одного модуля не требует изменения других модулей;
* Заменимость модулей – приведение системы в такое состояние, при котором модули могут заменять друг друга с сохранением работоспособности системы;
* Возможность тестирования – предоставление возможности отсоединить неисправный модуль от всех остальных, протестировать и исправить его работу;
* Переиспользование – возможность использования существующего модуля в других программах и другом окружении;
* Сопровождаемость – разбитую на модули программу легче сопровождать после вывода системы в эксплуатацию, улучшая её, оптимизируя и устраняя дефекты.

Построение модели программного средства – системная задача, требующая анализа и синтеза исходных данных, гипотез, теорий, знаний экспертов. Системный подход позволяет построить модель реальной системы и использовать эту модель для оценки системы по ряду показателей, например, эффективность управления или её функционирования. Приступая к разработке каждой программы, следует иметь ввиду, что она, как правило, является большой системой, поэтому нужно принять меры для её упрощения. Для этого такую программу разрабатывают по частям, которые называются программными модулями.

Каждый модуль состоит из спецификации и тела. Спецификации определяют правила использования модуля, а тело - способ реализации процесса обработки [8]. Принципы модульного программирования программных продуктов во многом схожи с принципами нисходящего проектирования. Изначально определяются состав и подчиненность функций, необходимых к реализации, а затем – набор программных модулей, отвечающих за выполнение этих функций.

При определении совокупностей модулей, реализующих функции конкретного алгоритма, необходимо учитывать следующее:

• каждый модуль вызывается вышестоящим модулем и, закончив свою работу, возвращает управление вызвавшему его компоненту;

• основные решения в алгоритме принимаются на максимально "высоком" по иерархии уровне;

• для выполнения одного и того же функционального блока в разных местах алгоритма вызывается один модуль, реализующий в себе необходимые для решения задачи шаги.

Среди множества модулей различают:

• головной модуль, управляющий запуском программной системы;

• управляющий модуль, обеспечивающий вызов других модулей на обработку;

• рабочие модули, которые выполняют функции обработки входных потоков данных;

• сервисные модули, библиотеки и утилиты, осуществляющие обслуживающие функции.

# 5. Применение конфигурационного механизма для адаптации сложной системы.

С точки зрения системно-комплексного подхода, конфигурация –качественная характеристика структуры, определяющая её пространственную, логическую и временную, а также комбинированную (комплексную) организацию сложной целостной системы [9].

Рассмотрим структуру сложной системы как совокупность компонентов, связанных между собой [10, с.8]:

S = <E, C, Ѱ, Пμ, Т> , где (1)

E – компонентный состав;

С – состав связей;

Ѱ – конфигурация;

Пμ – параметры, качественно и количественно характеризующие компоненты, связи и конфигурации;

Т – время.

Структура системы может изменяться при изменении входящих в неё компонентов, например, при добавлении новых элементов или при изменении связей уже между существующими частями. Формула (1) представляет собой структуру моноконфигурационной системы, характеризующуюся набором элементов E, связей между ними С, конфигурацией Ѱ, параметрами Пμ и временем Т. Для поликонфигурационных структур кортеж будет выглядеть следующим образом:

S = Ѱ1[Ѱ2, <E, C, Пμ, Т>] (2)

С точки зрения концептуального моделирования, конфигурация – это структура структур или метаструктура [11].

Тип конфигурации – это самостоятельный, отличающийся от всех других, способ системной организации структуры в единое целое. Каждая конкретная конфигурация характеризуется рядом существенных качественных и количественных свойств, т.е. атрибутов и их значений. К совокупности существенных факторов, характеризующих тип конфигурации в первую очередь, относят следующие:

1) физическая (материальная) или функциональная неоднородность, степень элемента, его ранг, положение в пространстве и состоянии во времени и т.п;

2) качественные особенности связей между элементами рассматриваемого типа конфигурации;

3) число элементов и связей, принадлежащих рассматриваемому типу конфигураций;

4) принципы, методы, способы организации элементов и связей в пространстве и во времени в единую целостную конфигурацию [9].

Пример сложной поликонфигурационной системы представлен на рис. 1.

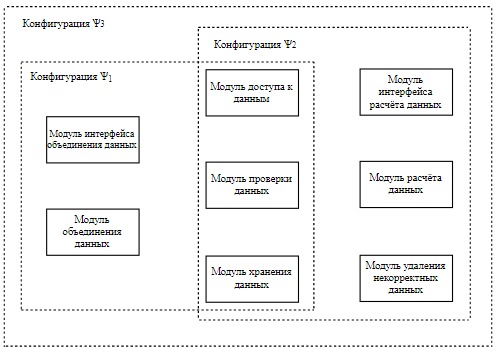


Рис.1. – Конфигурации сложной системы

Рассматриваемая система имеет 8 модулей, отвечающих за определённый круг задач. Различные наборы модулей в совокупности формируют три вида конфигураций, в рамках которых система может функционировать согласно решаемым задачам.

# 6. Конфигурационное моделирование

Любая конфигурация характеризуется строго определённой совокупностью свойств, она конечна и достаточно ограничена.

Введём в рассмотрение информационные объекты – описатели: признаки конфигураций, параметры конфигураций и показатели конфигураций.

Признак конфигураций – информационный объект, позволяющий узнать, определить или описать конкретное свойство конфигурации. Посредством признака определяются как качественные, так и количественные свойства конфигураций.

Параметр конфигураций – информационный объект, посредством которого осуществляется качественная или количественная оценка конкретного свойства конфигурации, представленного определённым признаком. Любой признак может быть определен конкретным набором качественных и количественных параметров.

Признаки и параметры конфигураций отражают строго определенные свойства конфигураций. На основе признаков и параметров может быть построен информационный массив, представляющий совокупность свойств конфигурационно-структурного разнообразия. Однако, такая совокупность не включает сведения, определяющие каждый конкретный тип конфигурации и вариант его реализации как единое целое. Ни отдельно взятые признаки, ни параметры не могут дать полного описания конфигурации.

Полное информационное описание конфигураций осуществляется посредством совокупностей признаков и наборов. Такие совокупности определяют показатель конфигурации.

Показатель конфигураций – комплексный информационный объект, представляемый посредством выделенных совокупностей признаков и наборов параметров, определяемых минимально необходимыми и достаточными качественными и количественными сведениями о конкретном типе конфигурации.

Иерархия типов описателей представлена на рис.2.

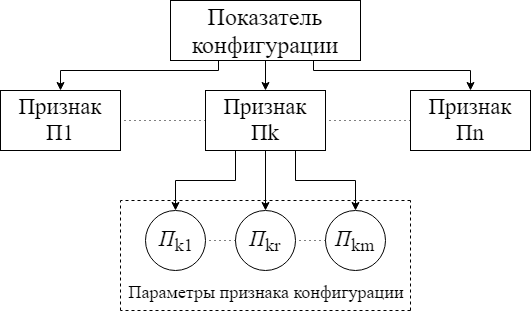


Рис.2. – Иерархия типов показателей конфигураций

Основные существенные признаки конфигураций отражают пространственную, логическую и временную организацию конфигураций.

Признак размерности пространства может принимать следующие значения:

1) одномерные – цепочечные конфигурации

2) двумерные – решетчатые конфигурации

3) трехмерные – кристаллические конфигурации

4) многомерные – гиперкристаллические конфигурации

Признак логической организации делится на пять уровней:

1) абсолютно централизованные структуры

2) централизованные

3) смешанные

4) децентрализованные

5) абсолютно децентрализованные

Признак организации конфигурации во времени может принимать следующие значения:

1) постоянные (неизменяемые)

2) переменные (перестраиваемые)

Таким образом, в зависимости от интерпретации описателей можно определить, как типы конфигураций, так и варианты их реализаций.

Задача конфигурационного моделирования структур состоит в том, чтобы на основе признаков конфигураций из множества типов конфигураций выделить один единственный, требуемый для конкретной решаемой задачи тип конфигурации.

# 7. Классификация задач конфигурационного моделирования

Сложные системы часто являются иерархическими и состоят из взаимозависимых подсистем, которые в свою очередь также могут быть разделены на подсистемы, и т.д., вплоть до самого низкого уровня [12]. Возьмём за базовую единицу конфигурации программный модуль. В рамках целой системы каждый из модулей выполняет определенную задачу. Взаимодействуя между собой, модули обеспечивают функционирование системы в рамках решаемых задач.

Представим задачи конфигурационного моделирования структур в виде древовидной иерархической организации [13]. Пусть самый верхний, первый, уровень содержит классы, второй – типы, третий – варианты задач моделирования структур (рис.3).



Рис.3. – Классификационная система задач конфигурационного моделирования

Классы задач определяются в зависимости от того, сколько различных типов конфигураций участвуют в формировании структуры. Решение задач первого класса – монокофигурационного – осуществляется в рамках определенного типа конфигураций. При решении поликонфигурационных задачах, в свою очередь, принимают участие два или более типов конфигураций. Структура может быть реализована либо на основе одного единственного варианта конфигурации определенного типа, либо совокупности различных вариантов в рамках одного и того же типа, или же различных типов конфигураций.

Типы задач выделяются в рамках соответствующих классов. Основанием для определения типов задач является факт участия или не участия в формировании структуры одного или нескольких вариантов реализации конфигураций рассматриваемого типа.

Варианты задач моделирования структур определены внутри каждого типа. В качестве основания для определения вариантов задач моделирования структур используется фактор кратности участия каждого из вариантов реализации конфигурации рассматриваемого типа в моделируемой структуре [9, с. 38-40].

# 8. Управление конфигурациями. Цели, задачи, основные действия и элементы

Управление конфигурациями – процесс, отвечающий за управление информацией о конфигурационных единицах, необходимой для предоставления ИТ-услуг. Конфигурационными единицами могут являться любые компоненты, которыми необходимо управлять с точки зрения жизненного цикла ИТ-услуги: это может быть, как аппаратное и программное обеспечение, так и документация и даже персонал.

Целями конфигурационного управления являются:

1) контроль: конфигурационное управление позволяет отслеживать изменения в контролируемых объектах, обеспечивает соблюдение процесса разработки;

2) управление: конфигурационное управление диктует процесс автоматической идентификации в ходе всего жизненного цикла ПО, обеспечивает простоту модификации и сопровождения ПО;

3) экономия ресурсов;

4) обеспечение качества.

Задачи конфигурационного управления:

1) идентификация конфигурации;

2) контроль конфигурации: контроль над изменениями материалов;

3) учёт текущего состояния: состояние документов, состояние кода, состояние отдельных задач и всего проекта в целом;

4) управление процессом разработки;

5) управление сборкой;

6) управление окружением;

7) отслеживание задач и проблем (в частности, отслеживание ошибок).

Основными действиям по управлению конфигурациями являются:

1) Сбор информации о каждом отдельном конфигурационном элементе;

2) Определение и анализ связей и взаимодействий между конфигурационными элементами;

3) Накопление информации в специальные базы данных управления конфигурациями, которые выступают в роли хранилища записей о конфигурациях на протяжении всего их жизненного цикла;

4) Контроль целостности системы после каждого изменения конфигураций;

5) Постоянный мониторинг ИТ инфраструктуры и её анализ.

Процесс конфигурационного управления дает логичную модель ИТ инфраструктуры и услуг. Он определяет, следит, обеспечивает и контролирует развитие различных конфигурационных элементов в инфраструктуре [14].

Во время формирования дисциплины управления конфигурацией в ней были воплощены следующие концепции:

1) Документы создаются для описания продукта и являются средством управления его конфигурацией;

2) Мониторинг изменений в продукте происходит посредством контроля изменений в документации;

3) Изменения в продукте не должны вступать в силу до тех пор, пока они не были отражены в документации;

4) До того, как быть отражены в документации и реализованы в самом продукте, изменения должны быть формально утверждены;

5) Все изменения должны отслеживаться;

6) Конфигурационные объекты, документы и их версии нумеруются и именуются единообразно и недвусмысленно;

7) Необходимо вести отчетность о состоянии изменений, документов и продуктов;

8) Каждый документ периодически сравнивается с соответствующим ему документом верхнего уровня на предмет выявления несоответствий;

9) Продукт в целом сравнивается со своим описанием (конфигурационной идентификацией) и должен этому описанию соответствовать.

Эти концепции были сгруппированы в четыре элемента управления конфигурацией, отображенных на рис.4:

* Конфигурационная идентификация (концепция 1)
* Контроль конфигурации (концепции 2, 3, 4, 5, 6)
* Учет состояния конфигурации (концепция 7)
* Ревизия и аудит конфигурации (концепции 8 и 9). [15]

Элементы управления конфигурацией

Рис. 4. Основные элементы управления конфигурацией

Первым элементом управления конфигурацией является конфигурационная идентификация. Она основывается на следующих составляющих:

* правила идентификации, определяющие что и каким образом идентифицируется;
* идентификация требований к продукту, отвечающая за способ идентификации требований к программной системе;
* идентификация изменений в данных определяет, как идентифицируются изменения в данных;
* базовые версии создаются для фиксации стабильных состояний программной системы и используются как возможные варианты её релизов;
* спецификации и диаграммы – документы, описывающие конфигурационную спецификацию программной системы;
* идентификация данных по релизам позволяет однозначно сопоставить элементы конфигурации и версии программной системы с её конкретным релизом.

Второй элемент управления конфигурацией – контроль конфигурации –

включает:

* критерии утверждения изменений, которые определяют формальные критерии, на основании которых принимается решение об утверждении или отклонении предложенных изменений;
* спецификации, модели, документация - это элементы конфигурации, подверженные изменениям и находящиеся в сфере действия контроля конфигурации;
* процедуры контроля конфигурации – утвержденные процедуры, которым должны следовать участники проекта;
* организация контроля изменений – организационная составляющая процесса, определяющая ответственность участников проекта при выполнении процедур контроля конфигурации.

Третьим элементом управления конфигурацией является учет состояния конфигурации. Он предполагает:

* ведение истории изменений конфигурации продукта. Благодаря ведению истории изменений, становится возможно определить кем, когда и какие именно изменения были сделаны;
* ведение истории состояний утвержденных изменений показывает, как менялись состояния утвержденных изменений от момента утверждения и до момента завершения их отработки;
* в истории верификации конфигурации хранятся данные о всех проведенных верификациях, а также их результаты;
* учет авторизации изменений указывает на то, кто отвечает за внесённые изменения.

Последним элементом управления конфигурацией является ревизия и аудит конфигурации. Он включает в себя:

* формальные квалификационные ревизии, определяющие соответствие элементов конфигурации предъявляемым к ним формальным требованиям;
* функциональный аудит конфигурации определяет соответствие конфигурации ПС функциональным требованиям, предъявляемым к продукту;
* физический аудит конфигурации – определяет наличие или отсутствие отдельных элементов в составе конфигурации.

Данные элементы управления конфигурацией позволяют учитывать все возможные изменения, происходящие в сложных системах, а вместе с тем и определять наилучшие способы их адаптации под решаемые задачи.

# Вывод

Таким образом, в данной работе рассматривалось применение конфигурационного моделирования как метода решения задачи адаптации сложных программных систем. Действительно, адаптивность, гибкость и масштабируемость программных комплексов являются острыми и актуальными проблемами на этапах их проектирования и реализации.

Конфигурационное моделирование позволяет создавать системы, способные к конфигурированию и реконфигурированию с помощью как внутреннего, так и внешнего компонента-конфигуратора.

Такой подход гарантирует наличие в сложной программной системе механизма адаптации под внутренние и внешние изменения путем конфигурации режимов работы системы под новые данные, полученные из внешней среды. Благодаря конфигуратору, программная система способна автоматически изменять своё состояние на целевое, удовлетворяющее новым требованиям, что резко сокращает расходы не только на создание и внедрение сложных программных систем, но и на поддержку и обслуживание в течение их эксплуатации. Возможность внедрения дополнительных модулей в уже существующую программную систему с дальнейшим их конфигурированием в совокупности с имеющимися модулями даёт большие перспективы в расширении функционала системы при минимальных затратах ресурсов.

**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. Л. А. Растригин. «Адаптация сложных систем» // Рига: Зинатне, 1981. — 375 с.

3. 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.

4. Х. Груневеген. Конфигурирование или адаптация? // Открытые системы. – 2008. – № 6 – С. 37-41

5. Создание модульных структур программного продукта – Студопедия [Электронный источник] http://studopedia.ru // URL: http://studopedia.ru/ 18\_64582 \_sozdanie-modulnih-struktur-programmnogo-produkta.html

6. Фаулер Мартин. Архитектура корпоративных программных приложений. Пер. с англ. – M. : Издательский дом «Вильямс», 2006. – 544 с.

7. Создание архитектуры программы или как проектировать табуретку – Хабрахабр [Электронный источник] https://habrahabr.ru // URL: https://habrahabr.ru/post/276593

8. Модульное программирование | Технологии программирования | Языки программирования Pascal и Delphi [Электронный источник] http://www.maksakov-sa.ru // URL: http://www.maksakov-sa.ru/TehProgram/ ModulProgram/index.html

9. Нечаев В.В. Конфигурационное моделирование: часть 1. Теоретические аспекты: Учебное пособие/ Государственное образовательное учреждение высшего профессионального образование «Московский государственный институт радиотехники, электроники и автоматики (технический университет)». – М.: 2007. – 92 с.

10. Нечаев В.В. Концептуальное метамоделирование структур. // Международная Академия информатизации; Отделение «Математическое и компьютерное моделирование». – М.: Международное изд. «Информация», 1997. – 52 с.; илл. 2; табл. 11; библ. 22 назв.

11. Большой словарь иностранных слов: - М.: ЮНВЕС, 1998. – 784 с.

12. Courtois, P. June 1985. On Time and Space Decomposition of Complex Structures. Communications of the ACM vol.28(6), p.596.

13. Нечаев В.В. Классификация задач синтеза структур в системах эволюционного моделирования. В кн.: Перспективы развития вычислительных систем (Применение идей эволюции и адаптации). II всесоюзный семинар (Рига, май 1985): РПИ, 1985. – с. 133 – 135.

14. Сайт по управлению услугами – Сервис менеджмент [Электронный источник] http://www.smlogic.ru // URL: http://www.smlogic.ru /g-it-s/itsm/protsess-upravleniya-konfiguratsiyami-configurat

15. Д. Лапыгин, А. Новичков. Конфигурационное управление проектами разработки программного обеспечения (2004). [Электронный источник] http://citforum.ru // URL: http://citforum.ru/SE/quality/configuration\_management