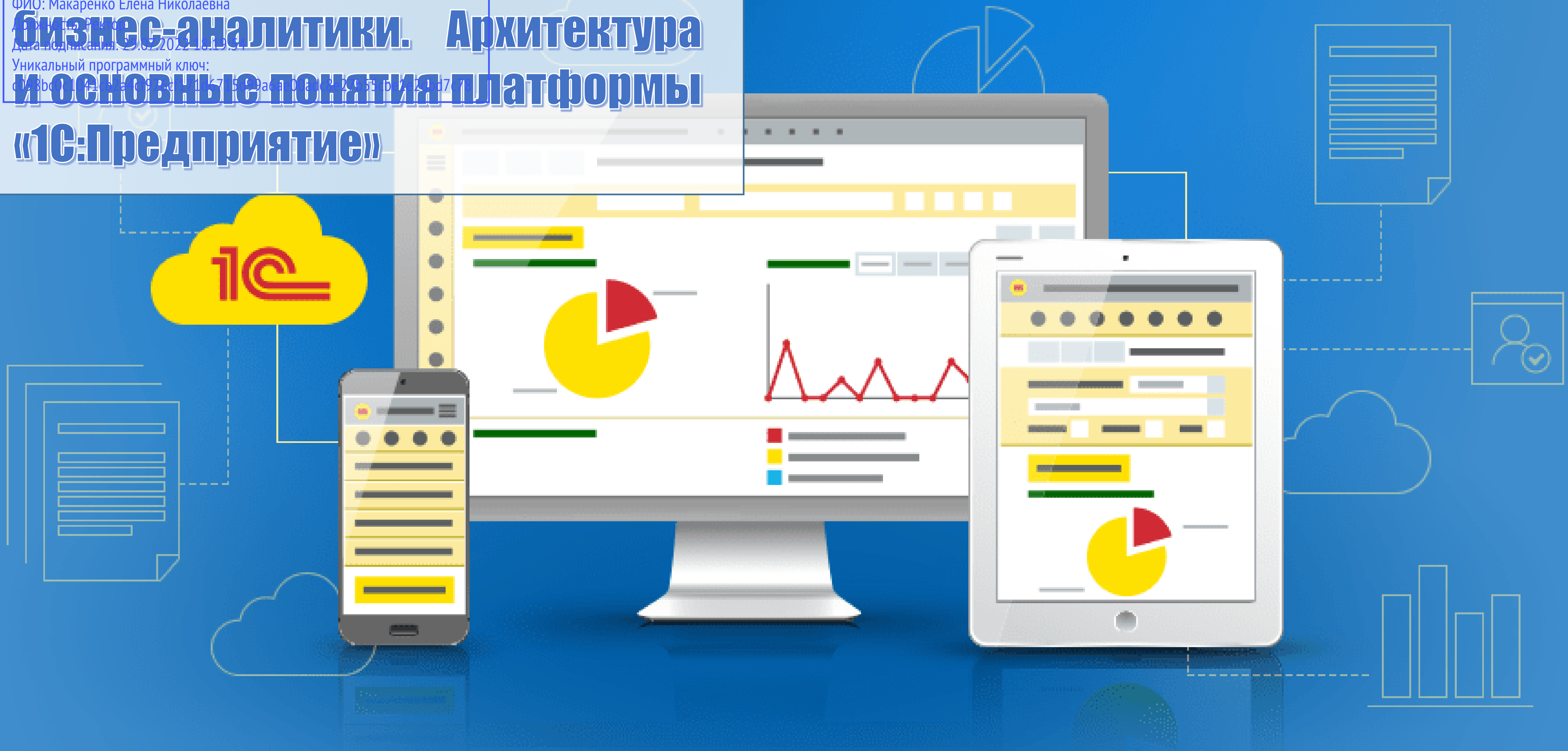
Тема 1. Архитектура систем бизнес-аналитики. Архитектура и основные понятия платформы «1С:Предприятие»



- *Архитектура платформы «1С:Предприятие».*
- *Метаданные как способ описания бизнес-приложения: построение прикладного решения на основе модели.*
- *Классификация объектов конфигурации.*
- *Варианты работы системы "1С:Предприятие": файловый вариант, клиент-серверный вариант.*
- *Виды взаимодействия компонентов: прямое подключение, подключение через веб-сервер, мобильная платформа.*
- *Виды клиентских приложений: тонкий клиент, веб-клиент, мобильный клиент, приложение на мобильной платформе, мобильный клиент с автономным режимом.*

Схема архитектуры решений на платформе «1С:Предприятие»

Платформа «1С:Предприятие» представляет собой совокупность двух составных частей: средства разработки и среда исполнения. Платформа обеспечивает работу конфигурации и позволяет вносить в нее изменения или создавать собственную конфигурацию. Технологии и механизмы платформы служат для создания, модификации и собственно функционирования конфигурации.

Помимо средства настройки типовых прикладных конфигураций, поставляемых фирмой «1С», платформа «1С:Предприятие» является и средством создания новых прикладных решений (как с использованием типовых конфигураций, так и без них). При этом оно применяется и для создания тиражных решений, и для разработки индивидуальных решений «под заказ».

Конфигурация – прикладное решение, предназначенное для автоматизации определенной области человеческой деятельности. Является самостоятельной сущностью и может выступать в качестве отдельного программного продукта. Но ее создание, изменение и «исполнение» осуществляется с использованием платформы.

Внедрения, как правило выполняются силами партнера и IT-специалистов заказчика с учетом особенностей деятельности предприятия и пожеланий заказчика.

Основная и конфигурация базы данных

Информационная база содержит: пользовательские ***данные*** + 2 конфигурации (***основная и конфигурация базы данных***) + административная информация+ *(необязательно) конфигурация поставщика*

Для чего же нужны эти две конфигурации?

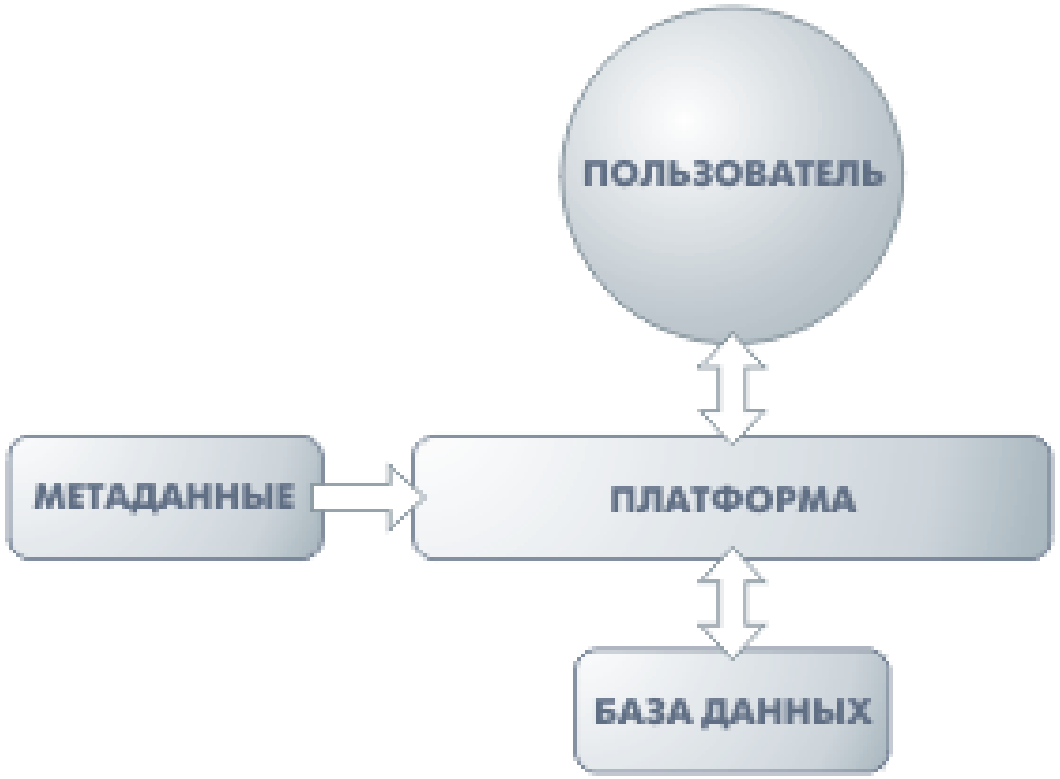
Разработчик работает именно с основной конфигурацией. То есть, когда разработчик вносит какие-либо изменения в конфигураторе, все изменения делаются именно в основной конфигурации.

С конфигурацией общей базы данных работают пользователи, они обращаются к ней и вносят изменения в данные информационной базы.

Для чего необходима такая схема?

Разработчик при такой схеме взаимодействия может менять основную конфигурацию, вносить в неё какие-либо изменения, а параллельно могут осуществлять работу операторы со своей конфигурацией. В тот момент, когда настанет необходимость синхронизации двух конфигураций, можно попросить пользователей выйти из информационной системы, когда разработчики будут готовы сделать обновление, и выполнить обновление конфигурации новой базы данных до основной конфигурации.

Модель метаданных как способ описания бизнес-приложения



В системе «1С:Предприятие» прикладное решение (конфигурация) не пишется в прямом смысле на языке программирования.

В основе бизнес-приложения лежат **метаданные**, которые представляют собой структурированное декларативное его описание и образуют иерархию объектов. Из них формируются все составные части прикладной системы и они определяют все аспекты ее поведения. Фактически, при работе бизнес-приложения платформа «проигрывает» (интерпретирует) метаданные, обеспечивая всю необходимую функциональность.

Объекты конфигурации

- это составные элементы, «детали», из которых складывается любое прикладное решение на платформе «1С:Предприятие»;
- представляют собой проблемно-ориентированные объекты, поддерживаемые на уровне технологической платформы. По большому счету задача разработчика заключается в том, чтобы собрать из этих объектов, как из конструктора, необходимую структуру прикладного решения и затем описать специфические алгоритмы функционирования и взаимодействия этих объектов, отличающиеся от их типового поведения;
- состав объектов, поддерживаемых технологической платформой, является результатом анализа предметных областей использования «1С:Предприятия», и выделения и классификации используемых в этих областях бизнес-сущностей. В результате этого анализа разработчик может оперировать такими объектами как справочники, документы, регистры сведений, планы счетов и пр. Разбивку объектов по видам можно увидеть в дереве конфигурации (они находятся на первом его уровне).

Основные группы объектов

Три основные группы объектов конфигурации:

– *Общие объекты* – вспомогательные объекты конфигурации, с помощью которых осуществляется создание конфигурации, механизмов взаимодействия пользователей с учетными данными.

– *Прикладные объекты*. Их перечень можно увидеть на первом уровне дерева метаданных (исключая ветку «Общие»).

– *Подчиненные объекты*. В зависимости от вида объекта конфигурации (прикладного или общего) он может иметь различные подчиненные группы объектов. К ним относятся «Реквизиты», «Табличные», «Реквизиты табличных частей», «Формы», «Макеты», «Графы», «Измерения», «Ресурсы» и т.д.

Взаимосвязь объектов конфигурации

Блок "Условно-постоянная информация"

- Содержит объекты, сохраняемые в базе данных и содержащие данные, меняющиеся сравнительно редко. Например, константа "Название организации", справочник "Сотрудники", перечисление "Тип клиента" и т.д.
- В этот блок данные вводятся один раз и используются много раз, в нескольких хозяйственных операциях, актах расчета.

Блок «Документы»

- включает, во-первых, документы, предназначенные для регистрации событий и операций, и, во-вторых, журналы, как средство их смысловой группировки. Например, документы «Приходная накладная», «Расходная накладная» и журнал «Складские документы». Документ характеризуется номером и датой;
- с помощью служебных объектов "Нумераторы" можно организовать "сквозную" нумерацию документов разных видов;
- служебный объект "Последовательность" предназначен для поддержания правильности движений по регистрам, путем строгого порядка проведения документов.

Блок «Регистры»

- предназначен для хранения информации о состояниях и количествах объектов базы данных, например, регистр сведений "Состояние сотрудников", "Цены товаров", регистры накопления "Продажи", "Остатки товаров" и т.д.
- кроме фактических данных, в регистрах могут храниться также плановые данные, например, плановый объем продаж, прогнозируемые курсы валют и т.д.

Блок «Обработка и вывод информации»

- Включает обработки и отчеты, которые используют уже введенные в базу данные для их обработки и представления пользователю (печати).
- Обработки предназначены для выполнения действий и расчетов над имеющейся в базе информацией, например, обработка «Закрытие периода».
- Отчеты формируют различные печатные формы, например, отчет "Анализ продаж".

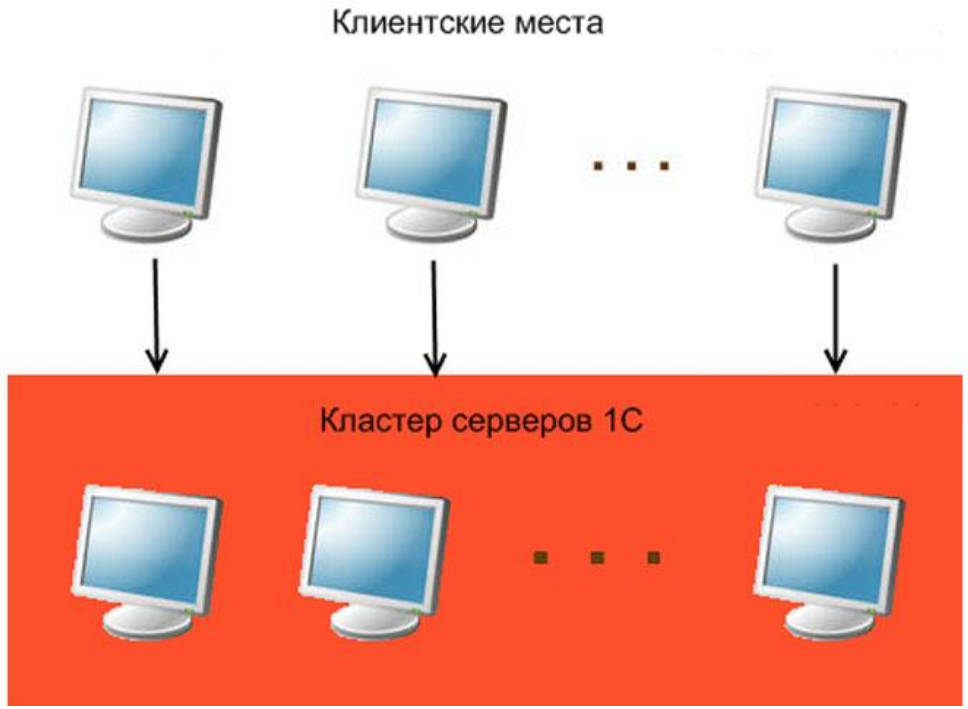
Файловый вариант работы

- рассчитан на персональную работу одного пользователя или работу небольшого количества пользователей в локальной сети;
- работу с базой данных осуществляет файловая СУБД, разработанная фирмой "1С" и являющаяся частью платформы;
- не требуются дополнительные программные средства, достаточно иметь операционную систему и системы «1С:Предприятие»;
- обеспечивается целостность информационной базы и простое создание резервных копий;
- все данные информационной базы (конфигурация, база данных, административная информация) располагаются в одном файле - файловой базе данных.

Клиент-серверный вариант работы

- предназначен для использования в рабочих группах или в масштабе предприятия. Он реализован на основе трехуровневой архитектуры «клиент-сервер»;
- разделяет всю работающую систему на три различные части, определенным образом взаимодействующие между собой: клиентское приложение, кластер серверов «1С:Предприятия», сервер базы данных;
- физически кластер серверов «1С:Предприятия 8» и сервер баз данных могут располагаться как на одном компьютере, так и на разных;
- использование кластера серверов «1С:Предприятия 8» позволяет сосредоточить на нем выполнение наиболее объемных операций по обработке данных;
- используются СУБД сторонних поставщиков. 1С работает со следующими СУБД: Microsoft SQL Server, PostgreSQL, IBM DB2, Oracle Database.

Кластер серверов

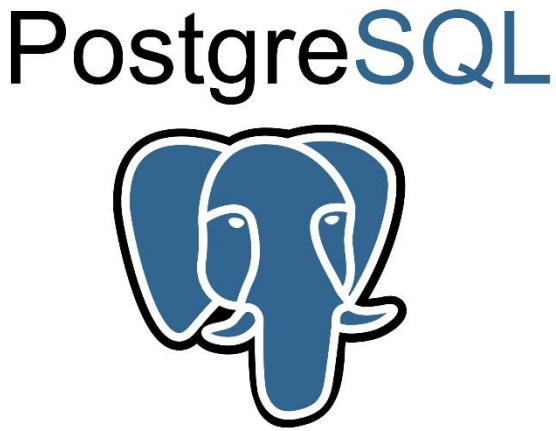


- основной компонент платформы, обеспечивающий взаимодействие между пользователями и системой управления базами данных в клиент-серверном варианте работы;
- позволяет обеспечить бесперебойную, отказоустойчивую, конкурентную работу большого количества пользователей с крупными информационными базами;
- является логическим понятием и представляет собой совокупность рабочих процессов, обслуживающих один и тот же набор информационных баз.

Сервер баз данных

В качестве сервера баз данных могут использоваться:

- Microsoft SQL Server,
- PostgreSQL,
- IBM DB2,
- Oracle Database.

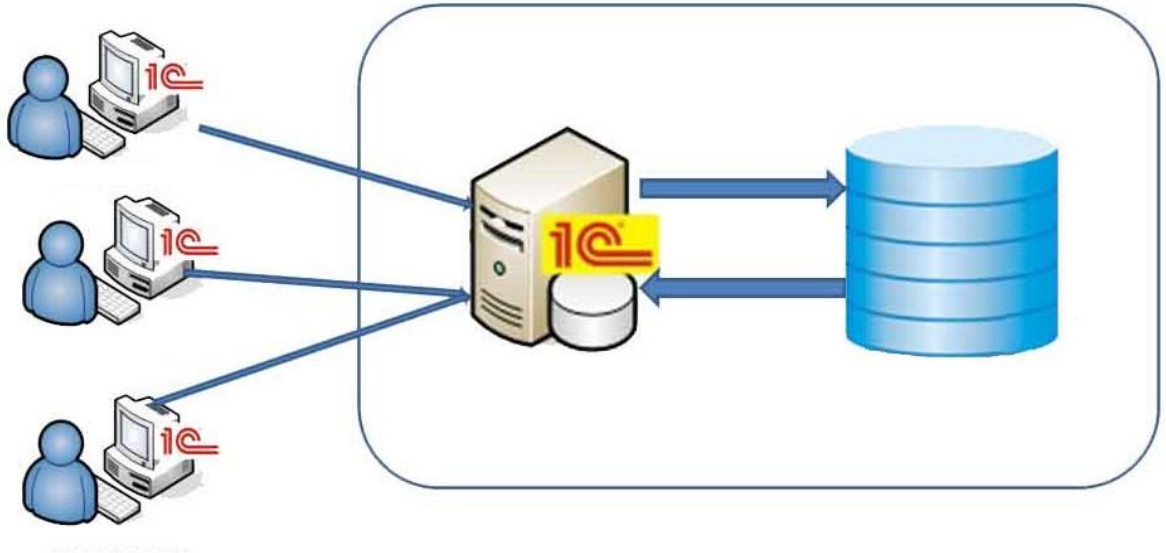


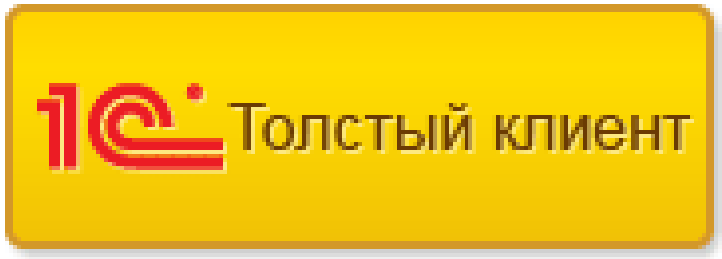
Клиентское приложение

Клиентское приложение – это программа, работающая на компьютере пользователя и обеспечивающая интерактивное взаимодействие системы 1С:Предприятие 8 с пользователем, в отличие от других компонент системы (программ и рабочих процессов), предназначенных исключительно для программного взаимодействия с другими частями системы или с другими программными объектами.

Виды клиентских приложений:

- толстый клиент ;
- тонкий клиент;
- веб-клиент;
- мобильный клиент.

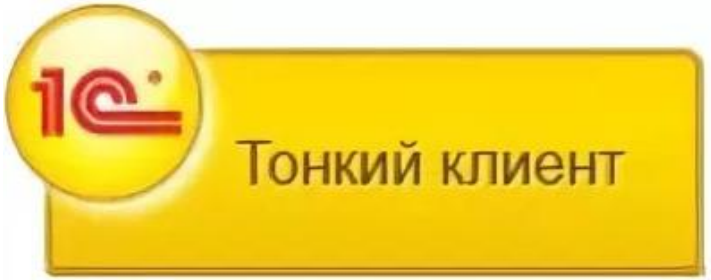




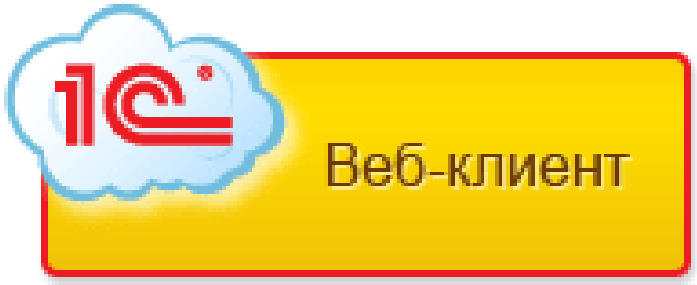
Толстый клиент

- это одно из клиентских приложений системы «1С:Предприятие 8». В операционной системе Windows исполняемый файл этого приложения — 1cv8.exe. В операционной системе Linux — 1cv8.;
- не поддерживает работу с информационными базами через интернет, требует предварительной установки на компьютер пользователя и имеет довольно внушительный объем дистрибутива;
- «толстым» клиент называется потому, что может исполнять практически всю функциональность, предоставляемую встроенным языком, в том числе умеет работать с прикладными типами данных, такими как **СправочникОбъект.<имя>, ДокументОбъект.<имя>** и т. д.

Тонкий клиент



- **Тонкий клиент** — это одно из клиентских приложений системы «1С:Предприятие 8». В операционной системе Windows исполняемый файл этого приложения — 1cv8c.exe. В операционной системе Linux — 1cv8c.
- не позволяет разрабатывать и администрировать прикладные решения, однако может работать с информационными базами через интернет;
- требует предварительной установки на компьютер пользователя, но имеет значительно меньший размер дистрибутива, чем толстый клиент;
- «тонким» клиент называется потому, что умеет исполнять ограниченный набор функциональности встроенного языка. В частности, на тонком клиенте недоступны все прикладные типы данных. Вместо этого тонкий клиент оперирует ограниченным набором типов встроенного языка, предназначенным лишь для отображения и изменения данных в памяти;
- тонкий клиент обеспечивает работу только в пользовательском режиме «1С:Предприятие». Режим работы Конфигуратор тонким клиентом не поддерживается.

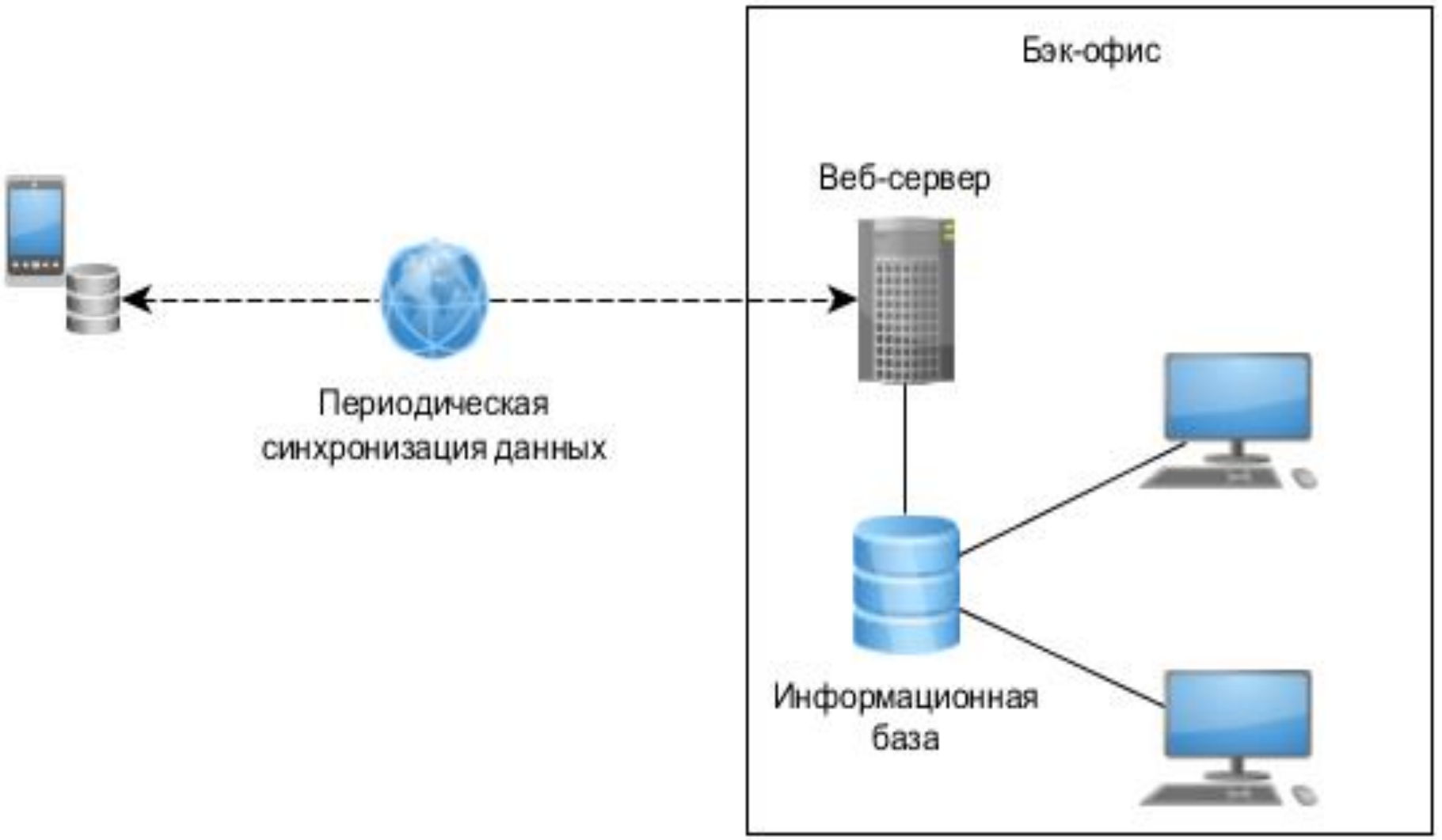


Веб-клиент

- не требует какой-либо предварительной установки на компьютер;
- в отличие от толстого и тонкого клиентов, исполняется не в среде операционной системы компьютера, а в среде интернет-браузера (Windows Internet Explorer, Mozilla Firefox, Google Chrome или Safari);
- веб-клиент использует технологии DHTML и HTTPRequest;
- Для работы в режиме веб-клиента требуется веб-сервер, настроенный на работу с «1С:Предприятием 8». Браузер клиента взаимодействует с веб-сервером по протоколу HTTP или HTTPS. Веб-сервер, в свою очередь, взаимодействует с «1С:Предприятием 8» в файловом или клиент-серверном варианте работы. В качестве веб-сервера используется Apache или IIS.

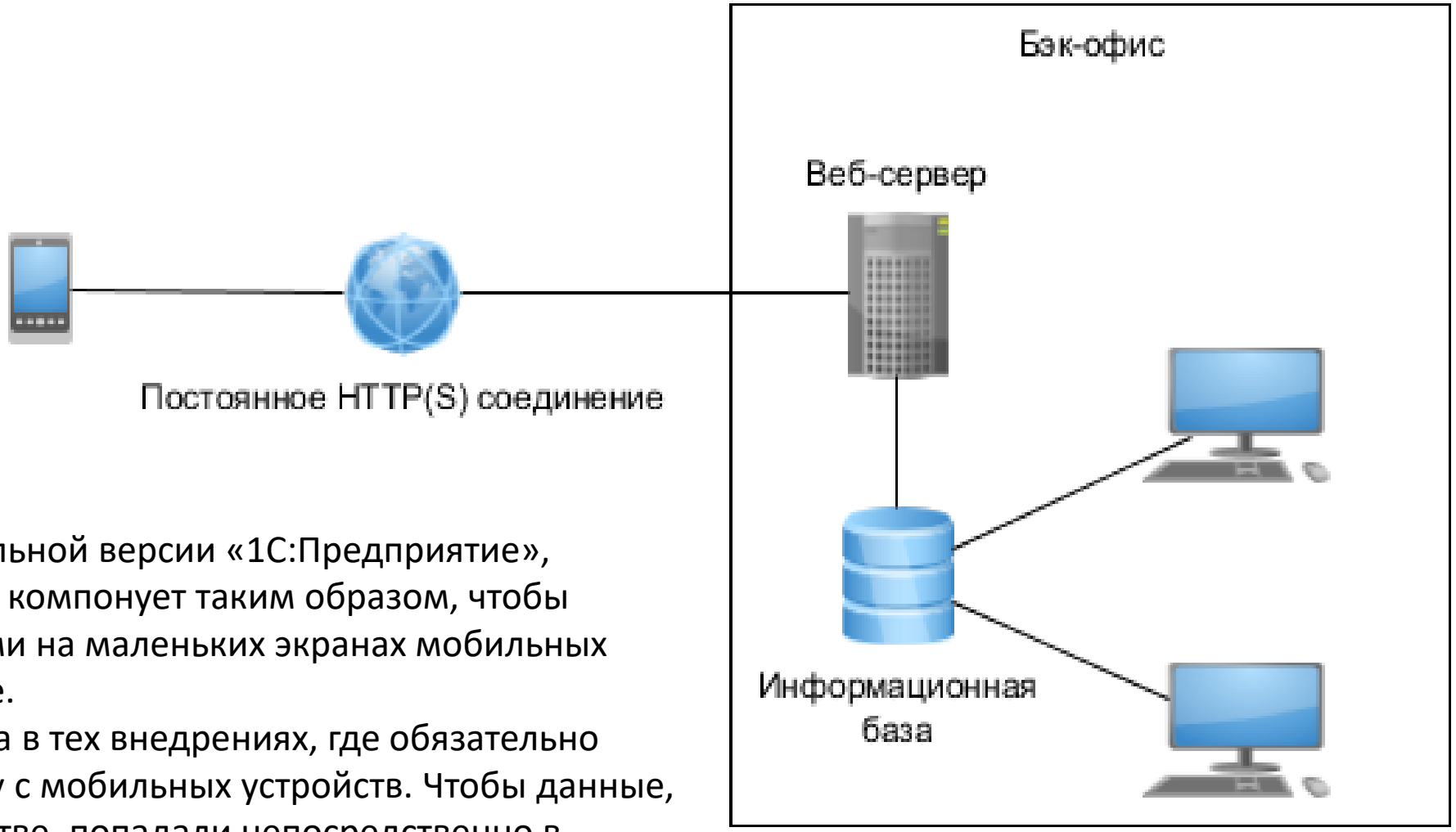
Мобильная платформа

По своей архитектуре такие приложения очень похожи на файловый вариант работы системы 1С:Предприятие. На мобильном устройстве существует собственная база данных, «внутри» мобильного приложения существует как клиент, обеспечивающий взаимодействие с пользователем, так и сервер, обеспечивающий взаимодействие с базой данных.
Такие мобильные приложения могут взаимодействовать с «основным» приложением, установленным в офисе. Но это не онлайн взаимодействие, а периодический обмен данными с бэк-офисом.



Мобильный клиент

Это тонкий клиент для мобильных устройств, который обладает интерфейсом, аналогичным мобильной платформе .
Мобильный клиент обеспечивает автоматическую трансформацию форм, декларативно описанных в конфигурации, в интерфейс, аналогичный интерфейсу мобильной платформы.



Формы, разработанные для настольной версии «1С:Предприятие», мобильный клиент автоматически компонует таким образом, чтобы обеспечить удобство работы с ними на маленьких экранах мобильных телефонов на приемлемом уровне.
Эта технология будет востребована в тех внедрениях, где обязательно требуется онлайн доступ в систему с мобильных устройств. Чтобы данные, введенные на мобильном устройстве, попадали непосредственно в «общую» базу данных, минуя промежуточные шаги синхронизации.

Различия клиентских приложений

	Толстый клиент	Тонкий клиент	Веб-клиент	Мобильный клиент
Работа с конфигуратором, разработка	Да	Нет	Нет	Нет
Работа в локальной сети	Да	Да	Да	Нет
Работа через Интернет	Нет	Да	Да	Да
Необходимость предварительной установки	Да, большой дистрибутив	Да, маленький дистрибутив	Нет	Да
Работа на мобильных устройствах	Нет	Нет	iPad	Да

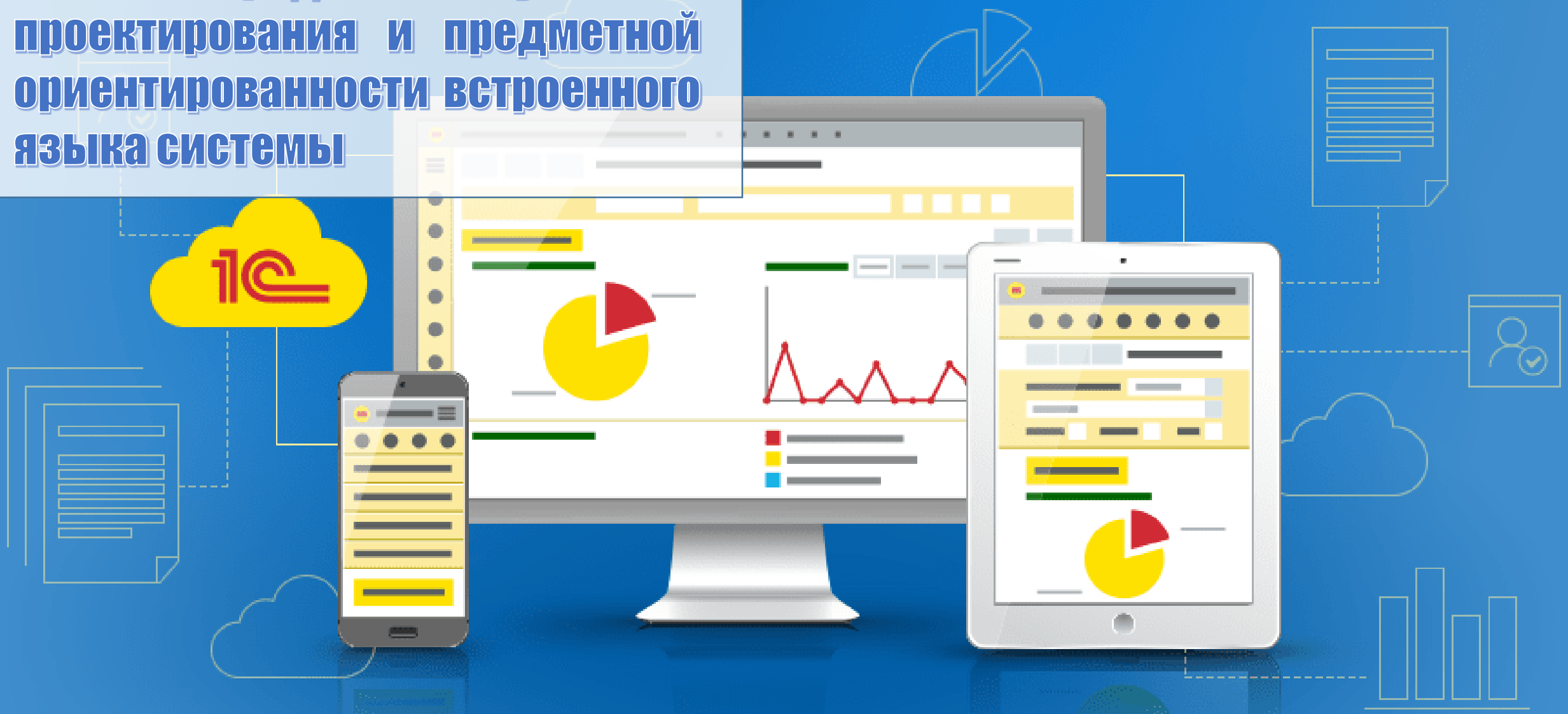
Конфигуратор

- **Конфигуратор** – один из двух режимов работы системы. Функционирование системы подразделяют на два разделенных во времени процесса: настройку (конфигурирование) и исполнение, исходя из чего система «1С: Предприятие» имеет два основных режима запуска – «Конфигуратор» и «1С:Предприятие» (перечисленные ранее виды клиентских приложений работают во втором режиме – режиме исполнения). Первый из этих режимов предназначен для создания информационной базы, внесения изменений в ее конфигурацию, а также для выполнения административных функций.

Режим «1С:Предприятие»

- В режиме «1С: Предприятие» пользователь запускает Конфигурацию на выполнение, как бы «проигрывая» файл информационной базы. При этом программная часть системы использует структуры, созданные на этапе конфигурирования, предоставляя пользователю возможность заполнить их конкретными значениями.
- Если на этапе конфигурирования с помощью встроенного языка определены соответствующие алгоритмы обработки, то в режиме «1С: Предприятие» пользователь будет вызывать их работу, давая системе соответствующие команды.

Тема 2. Парадигма визуального проектирования и предметной ориентированности встроенного языка системы



- *Парадигма визуального проектирования и предметной ориентированности встроенного языка системы.*
- *Управляемое приложение.*
- *Декларативное описание пользовательского интерфейса.*
- *Клиентская и серверная части приложения.*
- *Директивы компиляции.*
- *Типы модулей и их назначение.*
- *Основные конструкции встроенного языка.*

Технология «Управляемое приложение»

- *Самостоятельное формирование структуры формы и размещение полей платформой. Если раньше разработчики описывали положение поля, указывая пиксели, то теперь есть возможность лишь указать вид группировки;*
- *Форма состоит из реквизитов, представляющих данные формы, и команд – выполняемых процедур и функций;*
- *Код формы выполняется на стороне и сервера, и клиента. Ведь сама по себе форма – это объект конфигурации, создаваемый на сервере и отображаемый на клиенте. Значит, объединяет в себе клиентскую и серверную часть;*
- *На клиентской стороне стали недоступны многие типы данных и теперь отсутствует возможность изменить данные в информационной базе;*
- *Для каждой процедуры или функции должна быть указана специальная настройка – директива компиляции.*

Директивы компиляции

&НаКлиенте (&AtClient)

Директива определяет выполнение процедуры (функции) на клиенте. Используется на клиенте, доступны процедуры модуля и доступны данные форм.

&НаСервере (&AtServer)

Директива определяет выполнение процедуры (функции) на сервере. Выполняется на серверном приложении. В таких процедурах доступны данные формы, доступен серверный контекст формы и вызовы серверных процедур модуля. Данные формы передаются с клиента на сервер и обратно по окончанию вызова.

&НаСервереБезКонтекста (&AtServerNoContext)

Директива определяет выполнение процедуры (функции) на сервере вне контекста формы. В данном случае не будут доступен контекст формы и ее данные. Позволяет вызывать только внекотекстные процедуры и функции и не позволяет выполнять передачу данных между клиентом и сервером. Данный метод позволяет существенно снизить объем передоваемой информации.

&НаКлиентеНаСервереБезКонтекста (&AtClientAtServerNoContext)

Директива определяет выполнение процедуры (функции) на сервере и на клиенте, не имеющую доступа к данным формы, переменным. В данном методе имеется доступ к процедурам и функциям клиентских и серверных одновременно.

&НаКлиентеНаСервере (&AtClientAtServer)

Директива определяет выполнение процедуры (функции) на сервере и на клиенте. В данном методе имеется доступ к процедурам и функциям общих модулей – серверных, не глобальных и серверных и клиентских одновременно, не имеющую доступ к переменным.

Особенности встроенного языка 1С

- предварительная компиляция – перед исполнением модули, содержащие текст на встроенном языке, преобразуются во внутренний код;
- кэширование скомпилированных модулей в памяти;
- мягкая типизация – тип переменной определяется типом значения, которое она содержит, и может изменяться в процессе работы;
- отсутствие программного описания объектов конфигурации – разработчик может использовать либо встроенные в платформу объекты, либо объекты, созданные системой в результате визуального конструирования прикладного решения.

Некоторые базовые конструкции встроенного языка

Конструкция	Описание
`Перем Адрес;`	Явное объявление переменной. Адрес - имя переменной. Начинаться имя переменной должно с буквы, либо с символа подчеркивания. Имя может состоять из букв, цифр и символов подчеркивания.
`А = 3;`	Переменную во встроенном языке 1С явно можно не объявлять. Первое присвоение значения этой переменной инициирует ее создание системой

`// Чтобы написать`
`// комментарий -`
`// ставятся две косые`
	Подсказки, пометки разработчика, которые помогают разобраться или вспомнить логику работы программного кода, прочая служебная информация может быть оформлена в виде комментариев. Каждая новая строка комментария начинется с символов //.

`НашеЧисло = 3.7+13*4;`	Переменной **НашеЧисло** присваивается числовое значение. С данными числового типа можно выполнять арифметические операции: сложение, вычитание, умножение и деление. В качестве разделителя целой и дробной части используется точка!
`С = -0.765;`	Числовые значения могут быть отрицательными.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание
ЭтоСтрока = "вуз";	Переменной ЭтоСтрока присваиваем строковое значение. Значение строкового типа пишется в кавычках
ФИО="Петров" + " " + "Семен" + " " + "АЛександрович"; //результат: ФИО = "Петров Семен Александрович"	Сложение строк – конкатенация.
ДатаОтчета = '2021.03.30';	Переменной присваивается значение типа Дата, которое всегда записывается в одинарных кавычках.
ЧислоСекунд = '2021.01.15' – '2021.01.17'; // ЧислоСекунд = 86400	Над датами можно производить операцию вычитания, в результате получим разницу между датами, измеренную в секундах. В сутках 86 400 секунд (60 сек * 60 мин * 24 ч).
НоваяДата ='2021.09.23'+ 86400; //НоваяДата = '2021.09.24'	К дате можно прибавлять и вычитать число. В результате к дате либо прибавится, либо отнимется число секунд.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание

Процедура ИмяПроцедуры
(ИмяПараметра1,
ИмяПараметра2,...)
// текст комментария
// тело процедуры
...
КонецПроцедуры
	За ключевым словом Процедура идет имя, затем указывают имена параметров, заключенных в круглые скобки. Между словами Процедура и КонецПроцедуры записывается тело процедуры. Имя должно начинаться с буквы или символа подчеркивания. Порядок описания (следования) процедур и функций между собой значения не имеет.

Функция ИмяФункции(Имя Параметра1, ...) // тело функции Возврат(Возвращаемое значение) ; КонецФункции	За ключевым словом Функция идет имя, затем указывают имена параметров, заключенных в круглые скобки. Далее идет тело функции. Функция должна возвращать результат в место ее вызова. Тело завершается ключевым словом КонецФункции.
Конструкции, реализующие ветвление, циклы.	
Если Оценка >= 3 Тогда Результат = "Экзамен сдан!"; Иначе Результат = "Еще надо подучить!"; КонецЕсли;	Простое условие. После слова КонецЕсли должна быть точка с запятой, потому что так заканчивается оператор Если.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание
Результат = ?(Оценка >= 3, "Экзамен сдан!", "Еще надо подучить!");	Сокращенное Если. Краткая запись предыдущего простого условия.
Если (Доход > 100000) И (КодКатегории = 2) Тогда КонецЕсли;	Составное логическое выражение.
Если Оценка >= 4 Тогда Результат = "Экзамен сдан, хорошие знания!"; ИначеЕсли Оценка = 3 Тогда Результат = "Сдан, но знания слабые"; Иначе Результат = "Знаний нет!"; КонецЕсли;	Множественное условие. Если первое условие не выполняется, то проверяется второе. Если ни одно из условий не выполняется то выполняется блок Иначе.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание
Пока Номер <= 20 Цикл КонецЦикла;	Простой цикл с неизвестным числом повторений. После слова КонецЦикла должна быть точка с запятой, потому что так заканчивается оператор Пока.
Для Номер = 1 По 20 Цикл КонецЦикла;	Простой цикл Для (цикл с известным числом повторений).
Для каждого СтрокаТаблицы Из Таблицы Цикл КонецЦикла;	Разновидность цикла для циклического обхода коллекций значений (универсальных коллекций значений (массивы, таблицы значений и т.п.) табличных частей справочников, документов и т.д.). При каждой итерации цикла возвращается новый элемент коллекции. Обход осуществляется до тех пор, пока не будут перебраны все элементы коллекции.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание

Пока <условие> Цикл
  Если <условие> Тогда
  Продолжить;
  КонецЕсли;
КонецЦикла;
	Если необходимо передать управление в начало цикла, то используется оператор Продолжить.

Пока <условие> Цикл
  Если <условие> Тогда
  Прервать;
  КонецЕсли;
КонецЦикла;
	Если необходимо произвести досрочный выход из цикла, то используется оператор Прервать. В этом случае управление передается на операторы после цикла.

Наим=Стр.Наименование; Наим=Спр["Наименование"];	Два подхода, которые используются во встроенном языке при работе с объектными сущностями (объектами, с набором свойств, методов) для обращения к свойству объекта.
Спр.Печать();	Вызов методов объектов производится «через точку».

Пример структуры программного модуля

```
//*************** ОБЛАСТЬ ОБЪЯВЛЕНИЯ ПЕРЕМЕННЫХ *********************
Перем Фамилия Экспорт; //это глобальная переменная
Перем Имя, Отчество;     //это переменная модуля
Перем ФИО;       //это тоже переменная модуля и к ней можно обращаться
                 //из любой процедуры и функции нашего модуля

//************* ОБЛАСТЬ ОПИСАНИЯ ПРОЦЕДУР И ФУНКЦИЙ **************
Процедура Процедура1()
   Перем Итог;    //Итог это локальная переменная (переменная процедуры)
Итог = Фамилия+" "+Имя+" "+Отчество;

КонецПроцедуры
Функция Функция1()
   // операторы функции
Возврат(Фамилия + " "+ Имя);
КонецФункции
//****************** ОСНОВНОЙ ТЕКСТ ПРОГРАММЫ **********************
Фамилия ="Иванов";
Имя = "Иван";
Отчество = "Иванович";
//*****************************************************************************
```

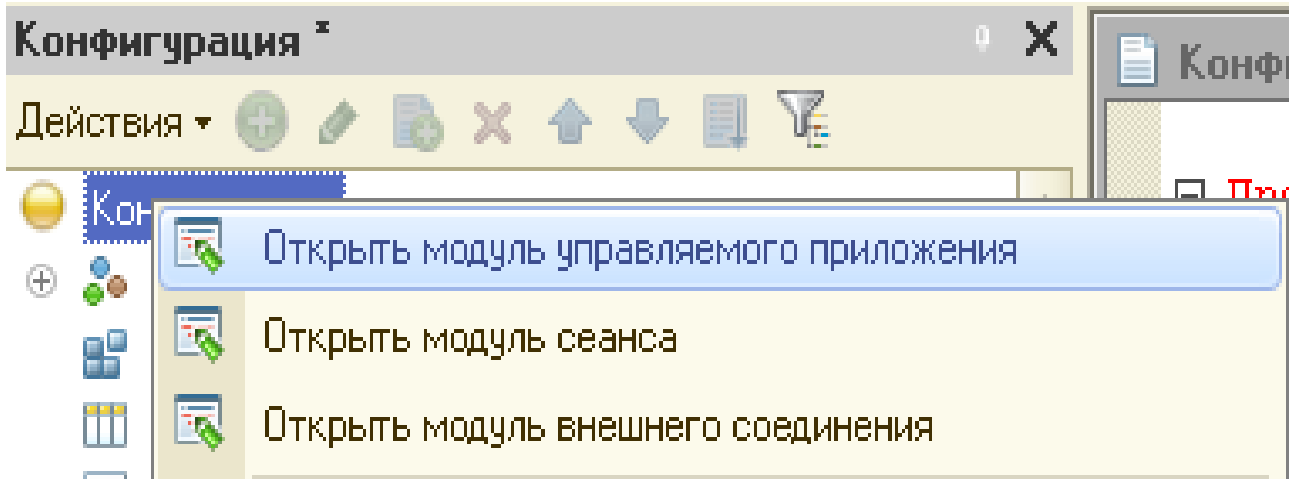
Типовая структура программного модуля включает:

- ***область объявления переменных.*** Размещается от начала текста модуля до первого оператора Процедура или оператора Функция или любого исполняемого оператора. В этом разделе могут находиться только операторы объявления переменных Перем;
- ***область описания процедур и функций.*** Размещается от первого оператора Процедура или оператора Функция до любого исполняемого оператора вне тела описания процедур или функций;
- ***основной текст программы.*** Размещается от первого исполняемого оператора вне тела процедур или функций до конца модуля. В этом разделе могут находиться только исполняемые операторы. Область основной текст программы исполняется в момент инициализации модуля. Обычно в разделе основной программы имеет смысл размещать операторы инициализации переменных какими-либо конкретными значениями, которые необходимо присвоить до первого вызова процедур или функций модуля

Виды модулей :

- Модуль приложения (управляемого или обычного);
- Модуль внешнего соединения
- Модуль сеанса
- Общие модули
- Модуль формы
- Модуль объекта
- Модуль менеджера объекта
- Модуль команды

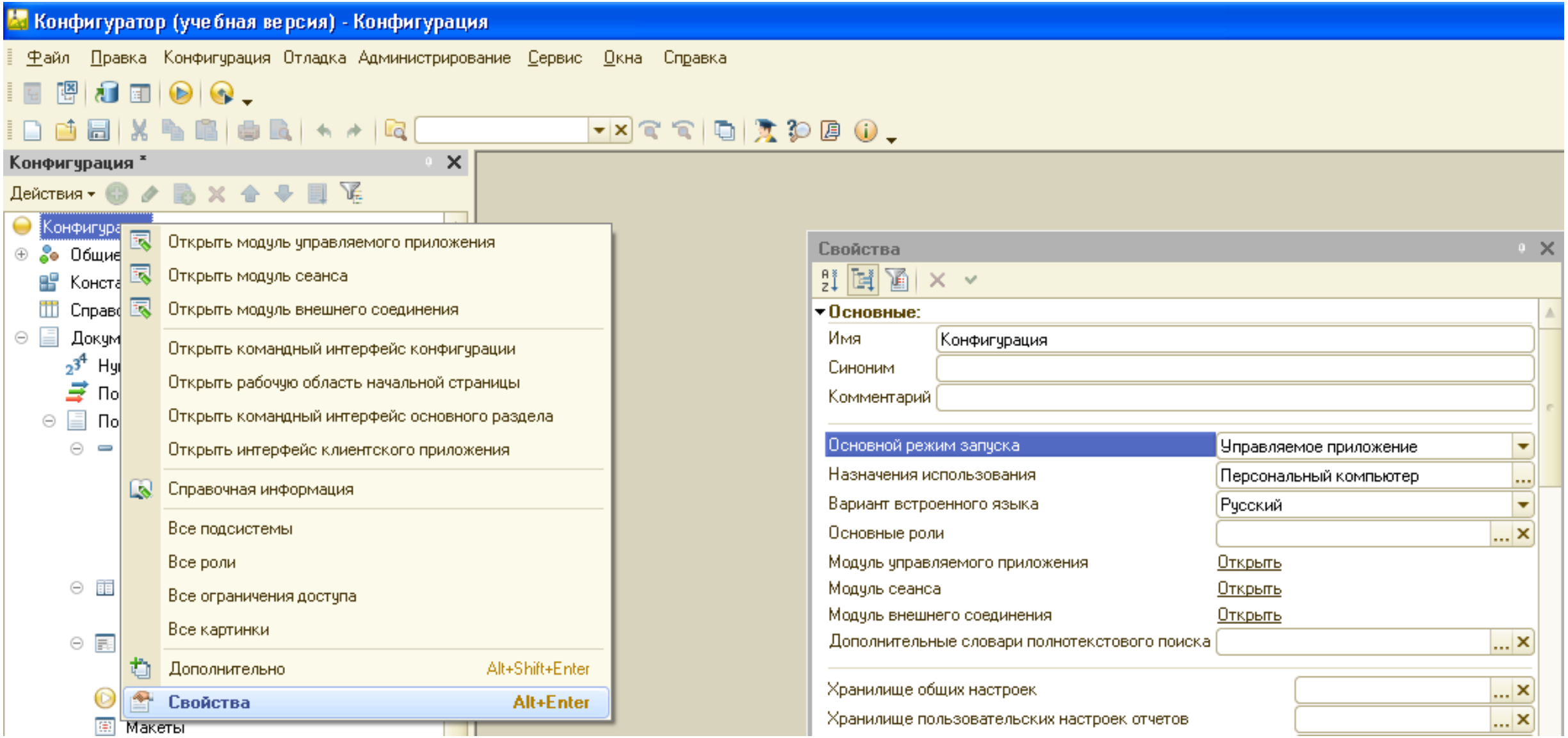
Модуль приложения (управляемого или обычного):



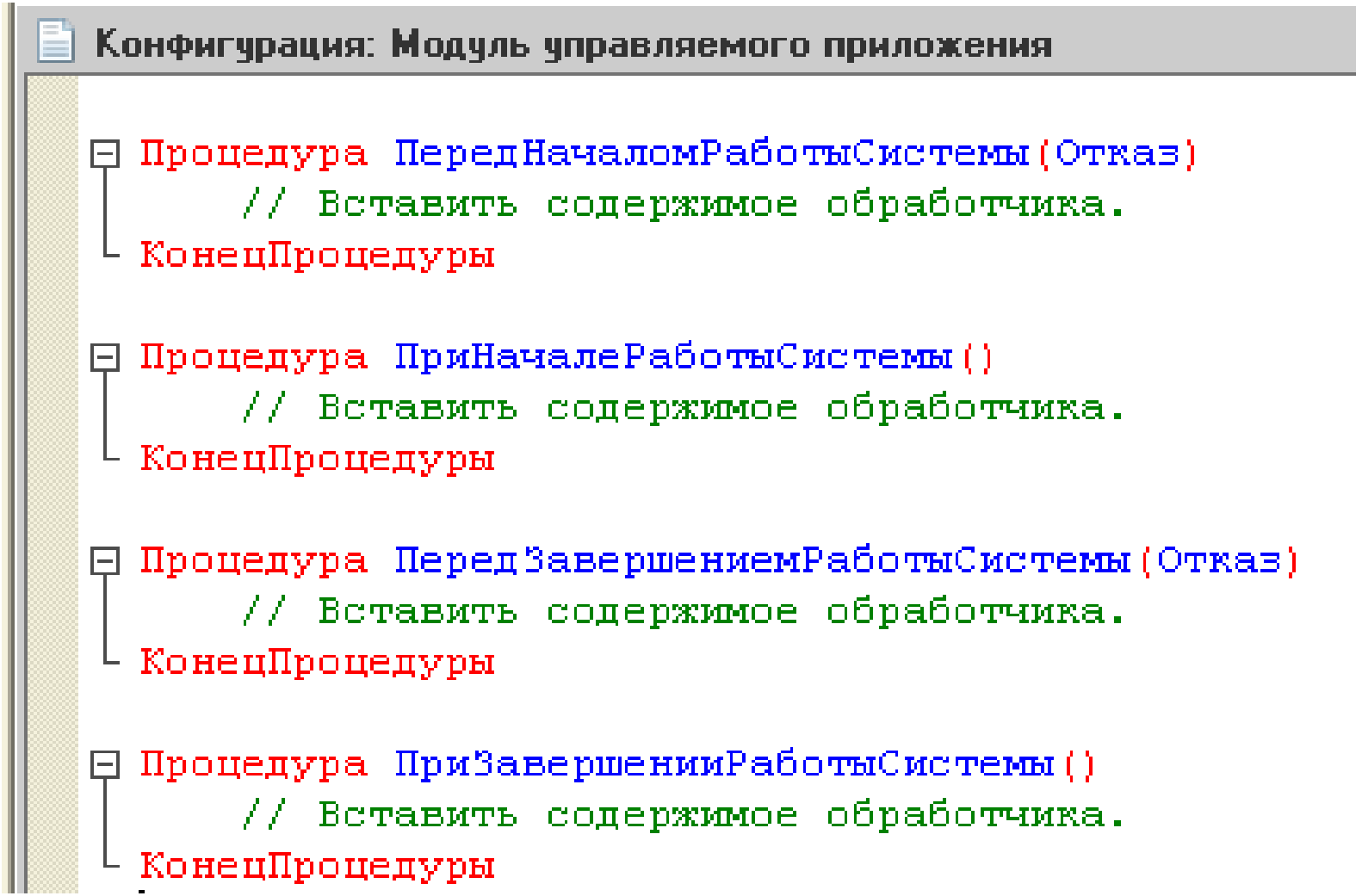
- может содержать все три области;
- выполняется на стороне клиента;
- располагается в корневом разделе конфигурации.

В модуле приложения описываются процедуры (обработчики) событий, которые инициализируются при старте и окончании работы системы. В платформе 1С 8 существует два различных модуля приложения: ***модуль Обычного приложения*** и ***модуль Управляемого приложения.*** Они срабатывают при запуске различных клиентов.

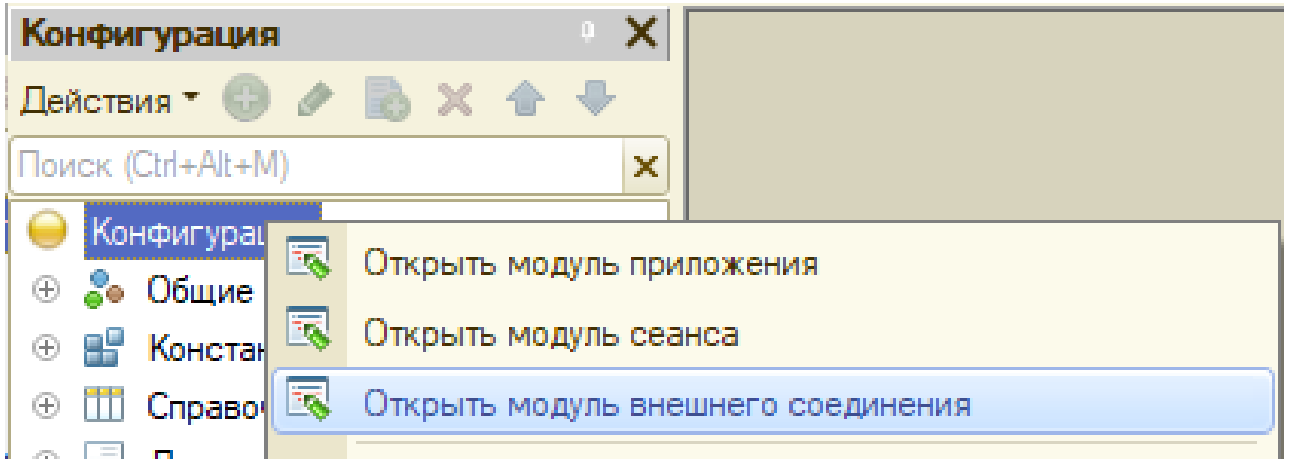
Настройка режима запуска приложения задается в свойстве конфигурации «Основной режим запуска»:



Модуль приложения обрабатывает события запуска и завершения приложения :



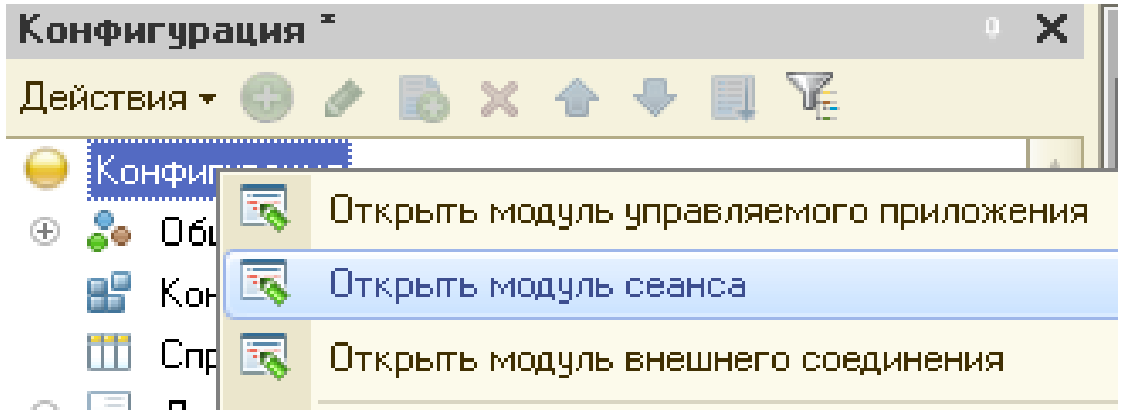
Модуль внешнего соединения:



- может содержать все три области;
- выполняется на стороне сервера;
- располагается в корневом разделе конфигурации.

В модуле внешнего соединения идет обработка событий старта и завершения работы приложения. Модуль внешнего соединения срабатывает, когда запуск приложения происходит в режиме com-соединения. Сам процесс внешнего соединения – это процесс не интерактивный. В этом режиме происходит программная работа с информационной базой и не происходит открытия окна приложения, что накладывает определенные ограничения на использование методов, предназначенных для интерактивной работы. В этом режиме нельзя использовать вызовы диалоговых форм, предупреждений и сообщений пользователю и т.п. Они просто не будут выполняться.

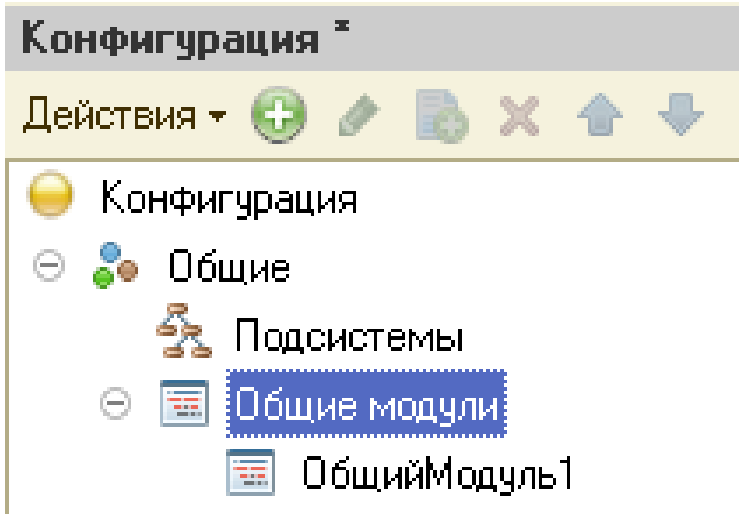
Модуль сеанса:



- может содержать область описания процедур и функций;
- выполняется на стороне сервера;
- располагается в корневом разделе конфигурации.

Это узкоспециализированный модуль, предназначенный исключительно для инициализации параметров сеанса. Его использование обусловлено тем, что само приложение может запускаться в различных режимах (что приводит к выполнению либо модуля управляемого, либо обычного приложения, либо модуля внешнего соединения), а инициализацию параметров сеанса необходимо производить вне зависимости от режима запуска. Чтобы не писать один и тот же программных код во всех трех указанных модулях, нам и потребовался дополнительный модуль, который выполняется вне зависимости от режима запуска приложения.

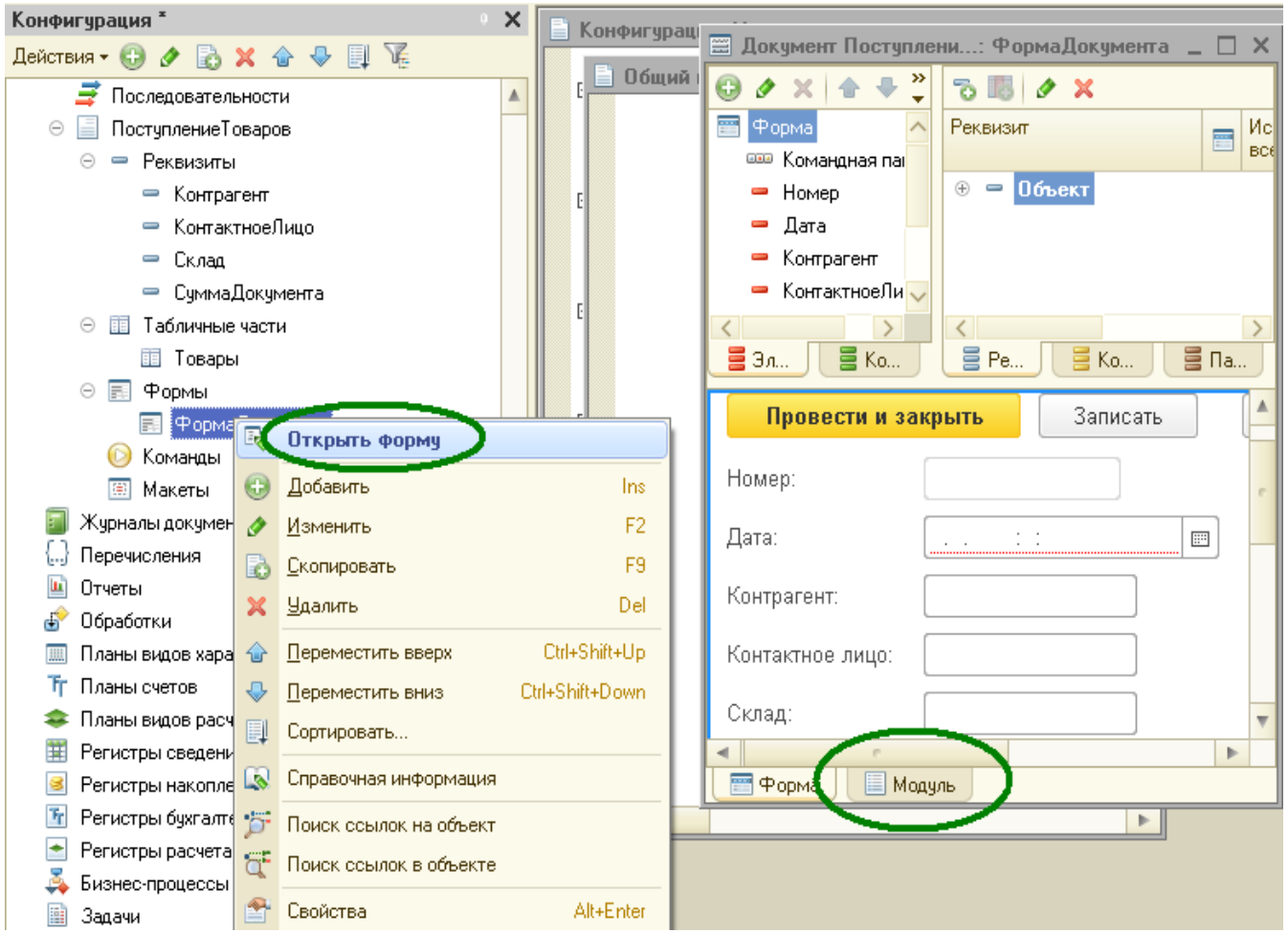
Общие модули:



- может содержать область описания процедур и функций;
- выполняется на стороне сервера или клиента (зависит от настроек модуля);
- располагается в ветке дерева объектов конфигурации «Общие» – «Общие модули»
- .

Общие модули предназначены для описания некоторых общих алгоритмов, которые будут вызываться из других модулей конфигурации. Общий модуль не содержит областей объявления переменных и основного текста программы. В нем можно объявлять экспортные методы, доступность которых будет определяться настройками модуля (на какой стороне он выполняется: на стороне сервера или клиента). В связи с тем, что раздел описания переменных не доступен, определять глобальные переменные в общих модулях нельзя. Для этого можно использовать модуль приложения.

Модуль формы:



- может содержать все 3 области;
- выполняется на стороне сервера и клиента.

Модуль формы предназначен для обработки действий пользователя с данной формой (обработка события нажатия кнопки, изменения реквизита формы и т.д.). Так же существуют события связанные непосредственно с самой формой (например, ее открытие или закрытие).

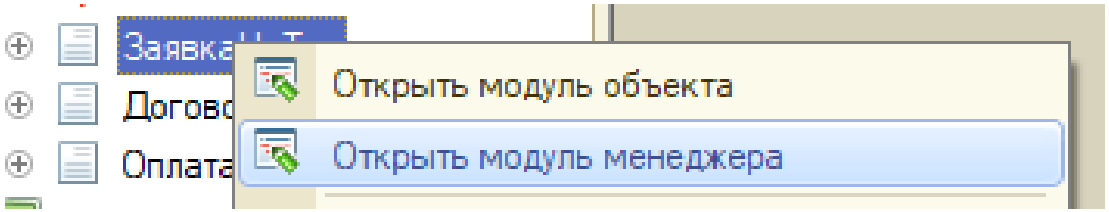
Модуль формы:

- может содержать все 3 области;
- выполняется на стороне сервера.

Данный модуль имеется у большинства объектов конфигурации и предназначен, в общем случае, для обработки событий, непосредственно связанных с объектом. Например, события записи и удаления объектов, проверка заполнения реквизитов объекта, проведение документа и т.д.

События модуля объекта будут вызываться в любом случае, даже в момент программной работы с объектом. Поэтому, если необходимо методы, связанные с объектом без привязки к конкретной форме объекта, то лучше использовать для этого модуль объекта.

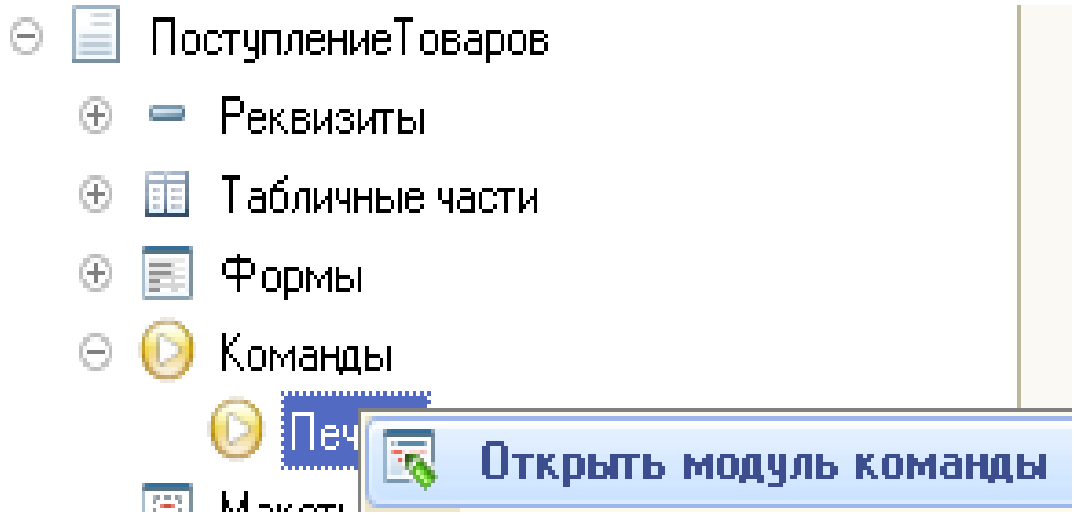
Модуль менеджера команды:



- может содержать все 3 области;
- выполняется на стороне сервера.

Модуль менеджера существует у всех прикладных объектов и предназначен для управления этим объектом как объектом конфигурации. Модуль менеджера позволяет расширить функциональность объекта за счет введения (написания) процедур и функций, которые относятся не к конкретному экземпляру объекта базы данных, а к самому объекту конфигурации. Модуль менеджера объектов позволяет размещать общие процедуры и функции для данного объекта и обращаться к ним из вне, например, из обработки (конечно, если эта процедура или функция будет с ключевым словом Экспорт).

Модуль команды:



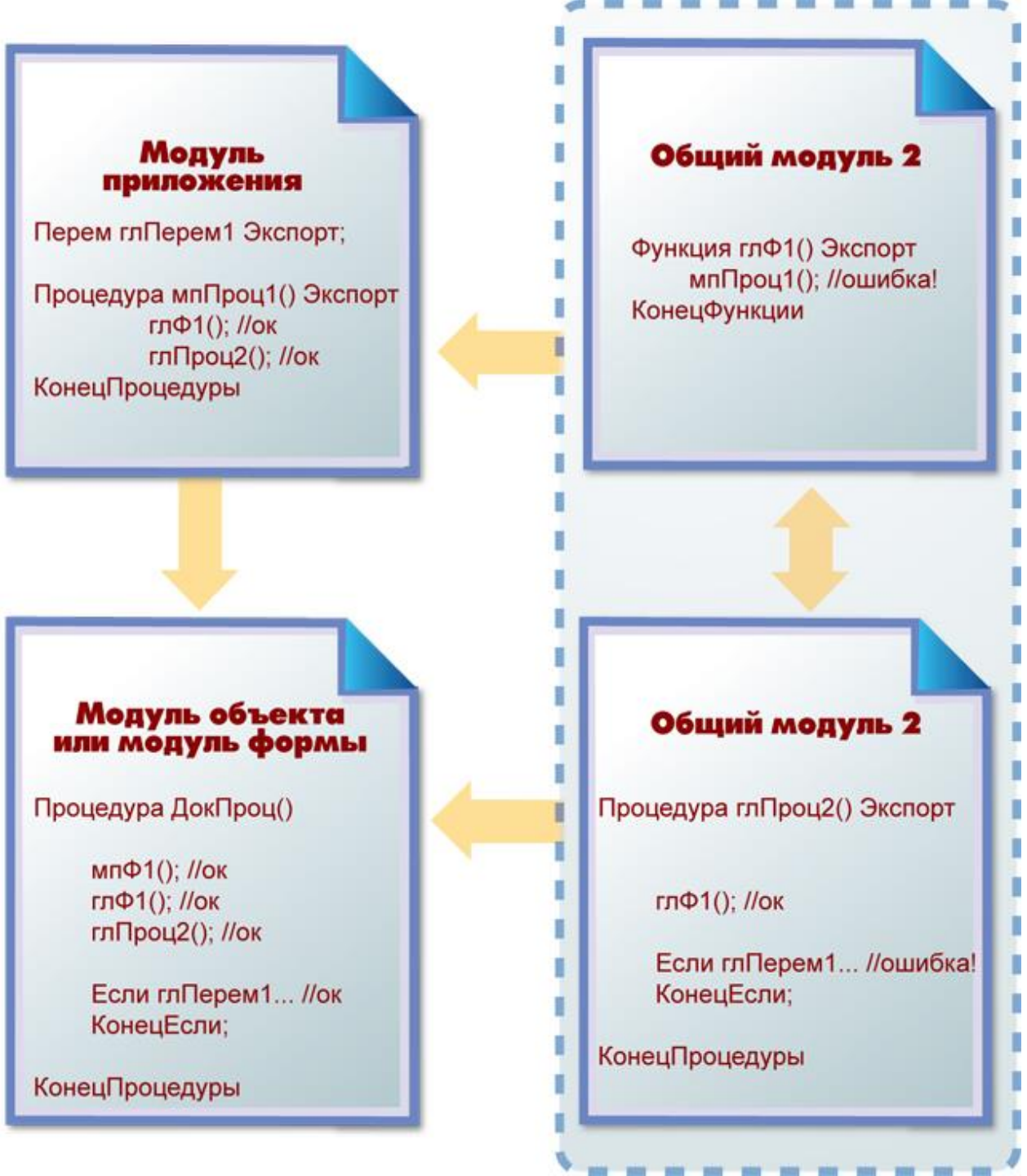
- может содержать раздел описания процедур и функций;
- выполняется на стороне клиента.

Команды – это объекты, подчиненные прикладным объектам или конфигурации в целом.

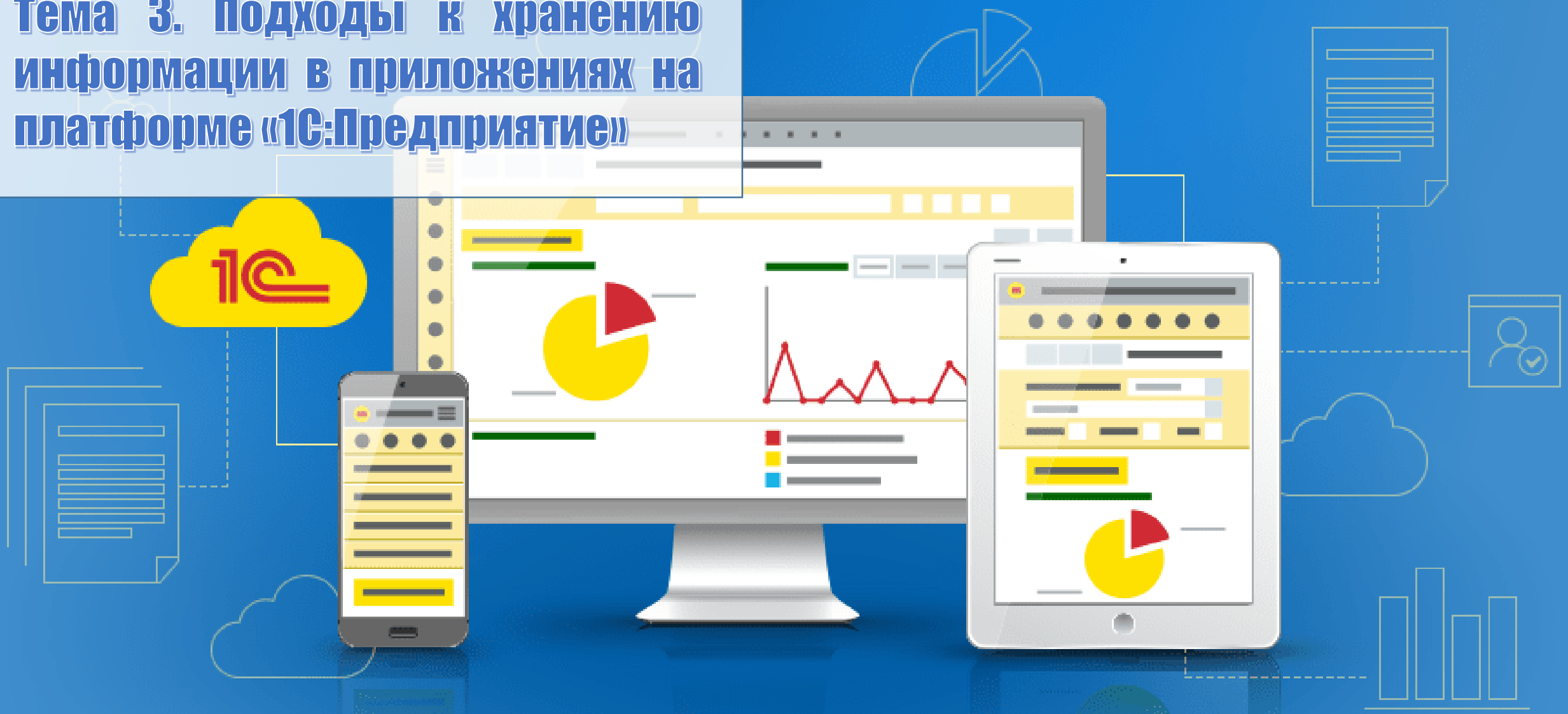
У каждой команды есть модуль команды, в котором можно описать предопределенную процедуру ОбработкаКоманды() для выполнения этой команды.

Правила видимости экспортируемых переменных, процедур и функций различных модулей:

1) В общем модуле недоступны экспортируемые переменные, процедуры и функции модуля приложения (модуля внешнего соединения).
2) В модуле приложения (модуле внешнего соединения) доступны экспортируемые процедуры и функции общих модулей.
3) В общих модулях доступны экспортируемые процедуры и функции других общих модулей.
4) В модулях прикладных объектов и модулях форм доступны экспортируемые переменные, процедуры и функции модуля приложения (модуля внешнего соединения), а также экспортируемые процедуры и функции общих модулей.
5) Если у формы назначен основной реквизит, то контекст модуля формы содержит дополнительные свойства и методы, связанные с основным реквизитом. Например, в модуле формы элемента справочника Номенклатура доступны свойства и методы объекта СправочникОбъект.Номенклатура.



Тема 3. Подходы к хранению информации в приложениях на платформе «1С:Предприятие»



- *Подходы к хранению информации в приложениях на платформе «1С:Предприятие».*
- *Хранение информации объектных и необъектных сущностей.*
- *Хранение иерархической информации.*
- *Хранение информации, имеющей привязку ко времени.*
- *Хранение вспомогательной информации (в объекте «ХранилищеЗначений») и др.*

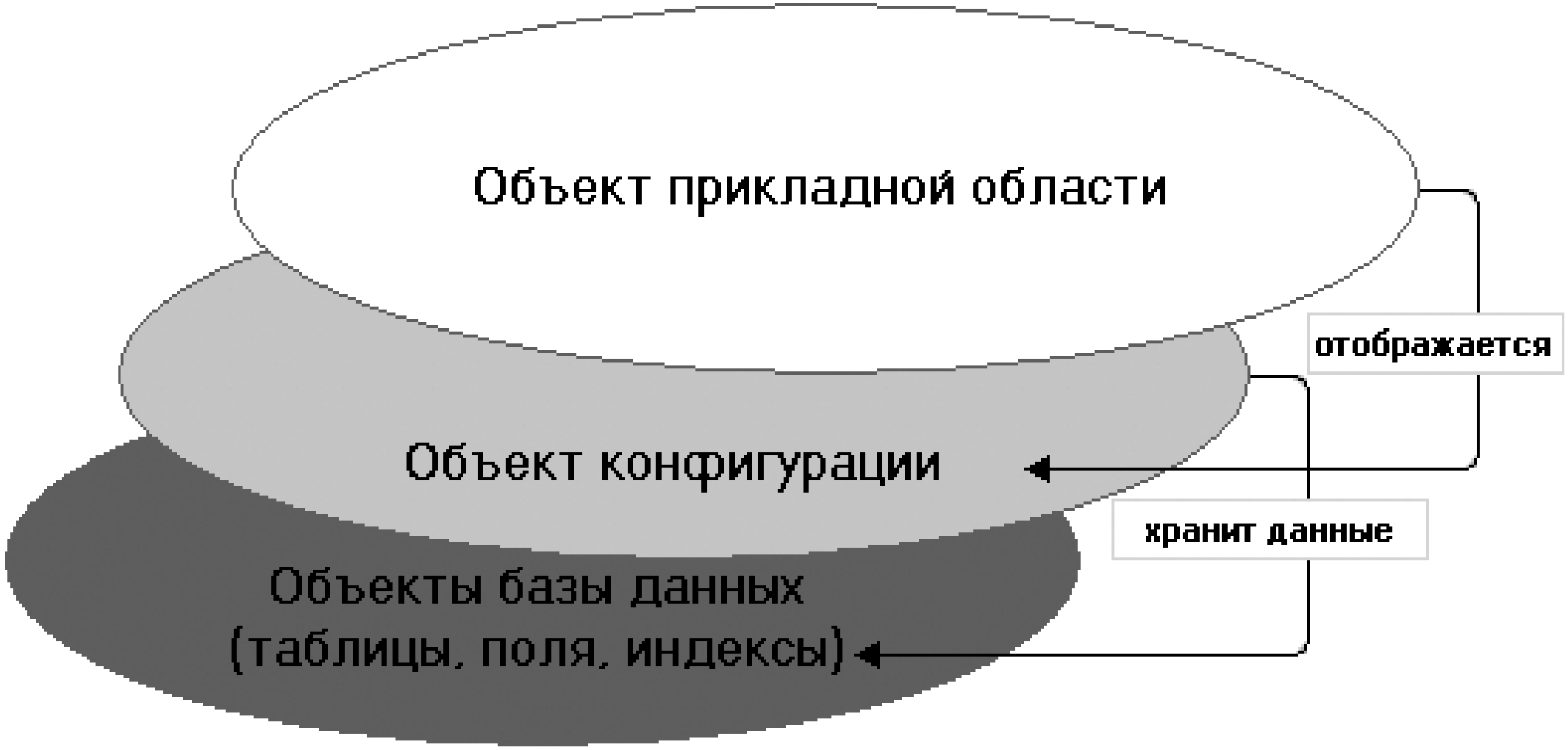
Противоречия между требованиями к хранимой информации

- необходимость обеспечения удобства представления логики взаимодействия сущностей в информационной модели;
- необходимость хранения больших объемов информации;
- повышенные требования к широкой функциональности и высокой производительности доступа к этим данным.

Противоречия между требованиями к хранимой информации

Чем больше объем, тем в общем случае сложнее с обеспечением скорости доступа к информации

Объектно-реляционная парадигма системы «1С:Предприятие»



В рамках средств платформы для каждого прототипа уже предопределены:

- оптимальная для большинства задач структура хранения информации в реляционной базе данных;
- набор средств встроенного языка для манипулирования этой информацией;
- методы, свойства, события и типовые для решаемых задач операции;
- способы отображения и редактирования;
- средства регулирования прав доступа и т. д.

Хранение информации объектных и необъектных сущностей

- К объектным данным относятся данные справочников, документов, планов видов характеристик, планов счетов, планов видов расчета, бизнес-процессов, задач, планов обмена.
- К необъектным данным относятся данные регистров сведений, регистров накопления, регистров бухгалтерии, регистров расчета, перерасчетов, последовательностей и констант.

При выборе прототипов объектов для хранения информации одним из типичных вопросов, возникающих при неочевидных случаях, является выбор между объектными и необъектными данными. Например, между регистром сведений и справочником.

Хранение информации объектных и необъектных сущностей

Справочник "Сотрудники"			
Ссылка	Код	Наименование	...

Справочник "ФизическиеЛица"			
Ссылка	Код	Наименование	...

Для принятия решения рекомендуется обращать внимание на природу данных предметной области. Если они обладают некоей «сутью», несмотря на смену значений отдельных свойств этого объекта, то предпочтение следует отдавать определению этих данных как объектных.

Однако следует учитывать, что выбор вида объекта конфигурации не должен производиться для каждой сущности отдельно. Необходимо анализировать наличие и остальных сущностей в комплексе всей прикладной задачи. Причем желательно не только на текущий момент, но и с учетом развития задачи в будущем.

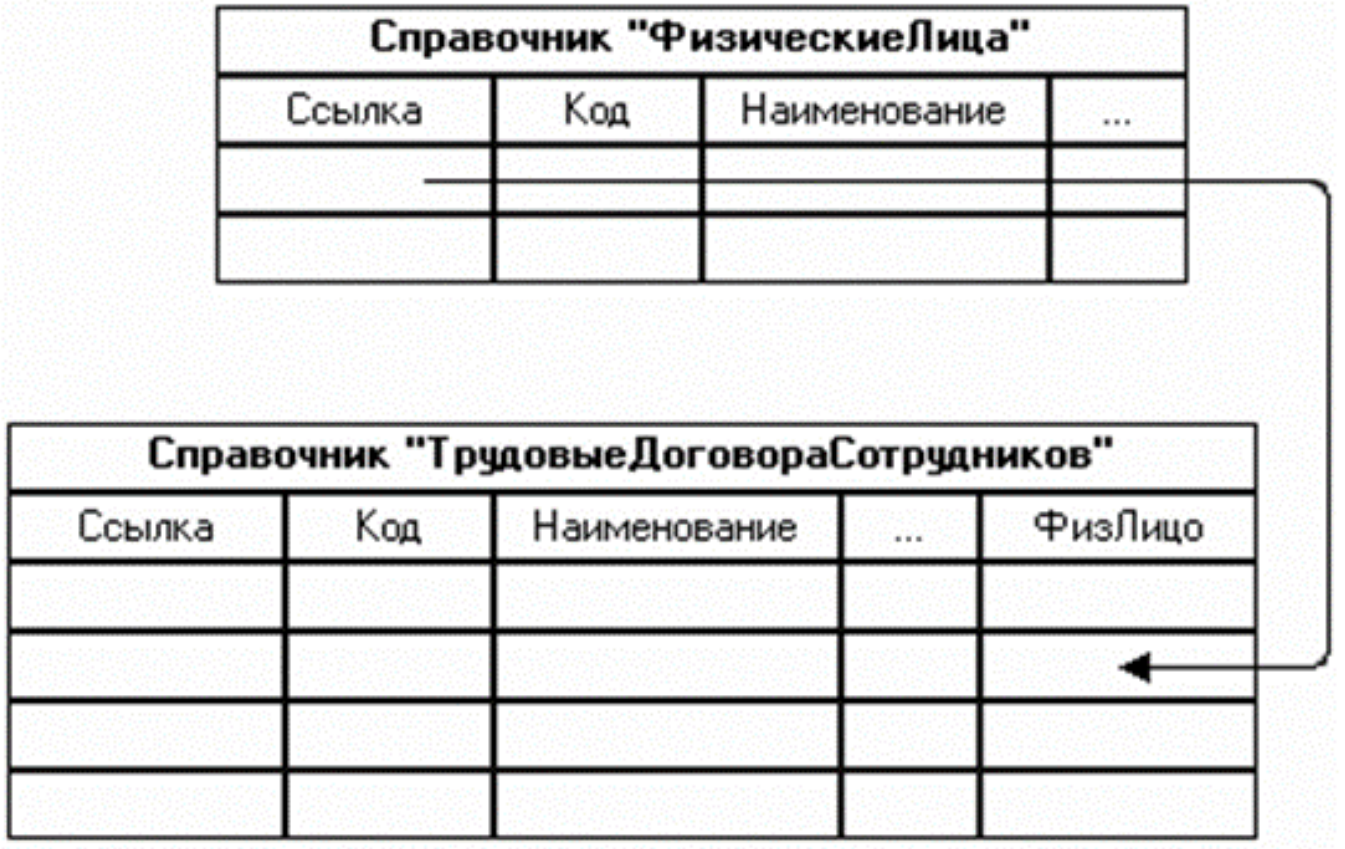
Регистр сведений "СотрудникиПодразделений"		
ФизЛицо	Подразделение	Должность

Хранение информации объектных и необъектных сущностей

Хранение данных о сотрудниках и их трудовых договорах - можно также выделить две сущности:

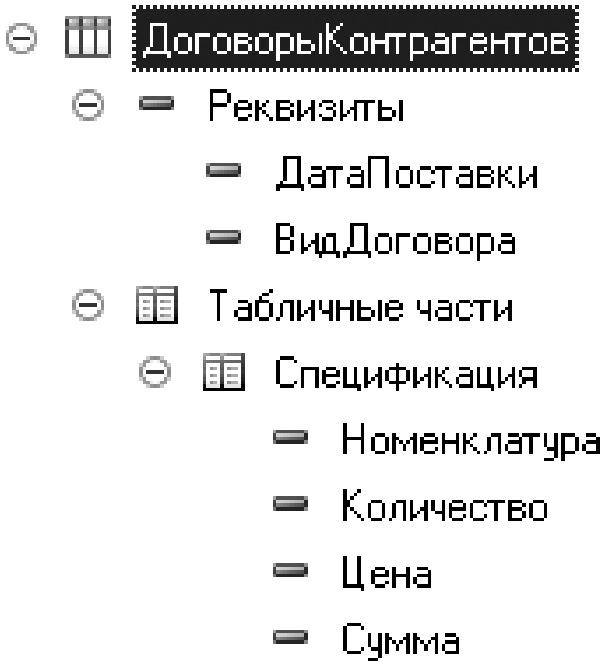
- «***Физическое лицо***»;
- «***Сотрудник – трудовой договор***» как отражение юридических отношений данного лица с организацией, имеющее при этом объектную природу, поскольку впоследствии ссылка на трудовой договор будет фигурировать во многих документах и регламентированных отчетах.

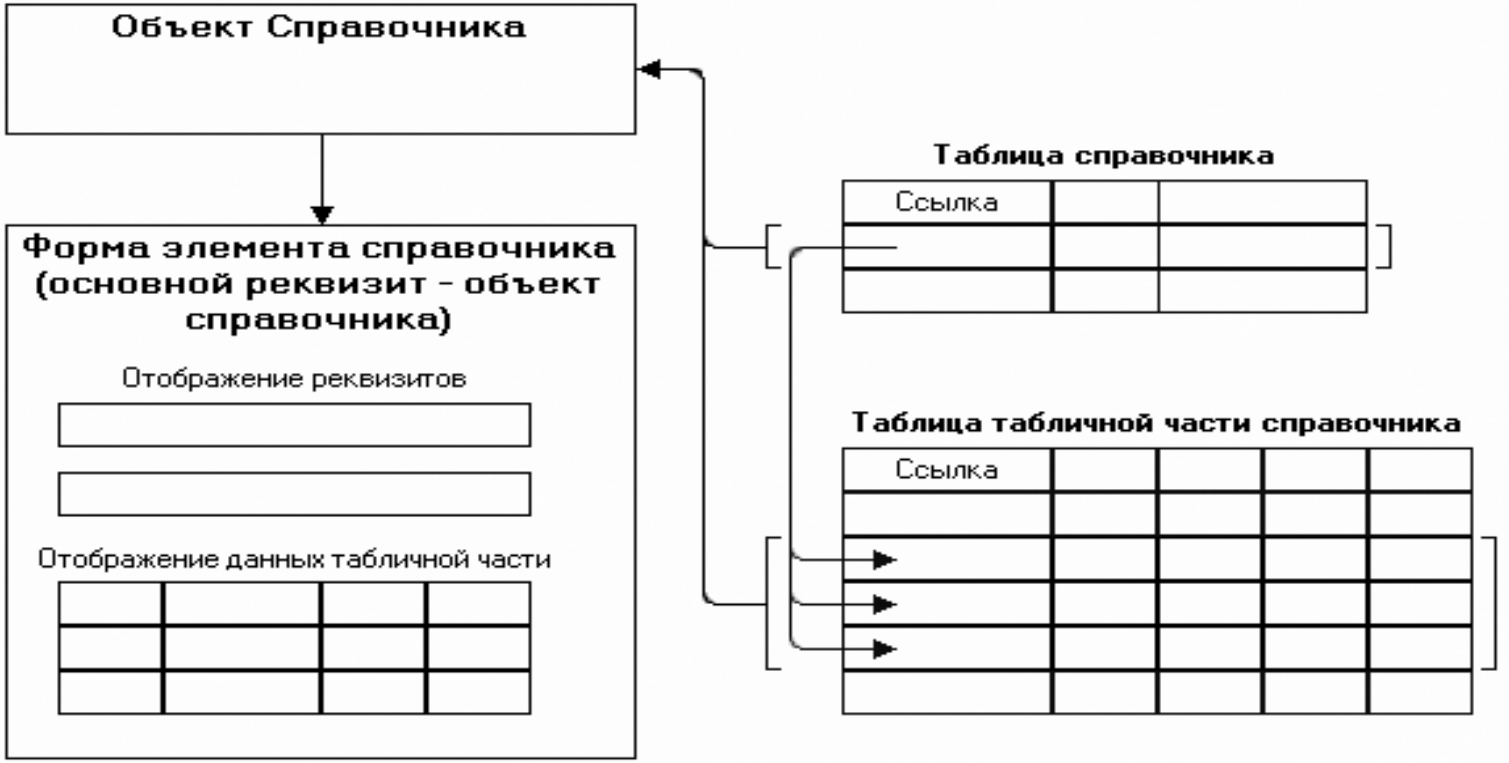
В этом случае для хранения данных следует использовать два справочника, один из которых (справочник ТрудовыеДоговораСотрудников) будет подчинен другому (справочник ФизическиеЛица)



Взаимосвязь справочника с подчиненным справочником

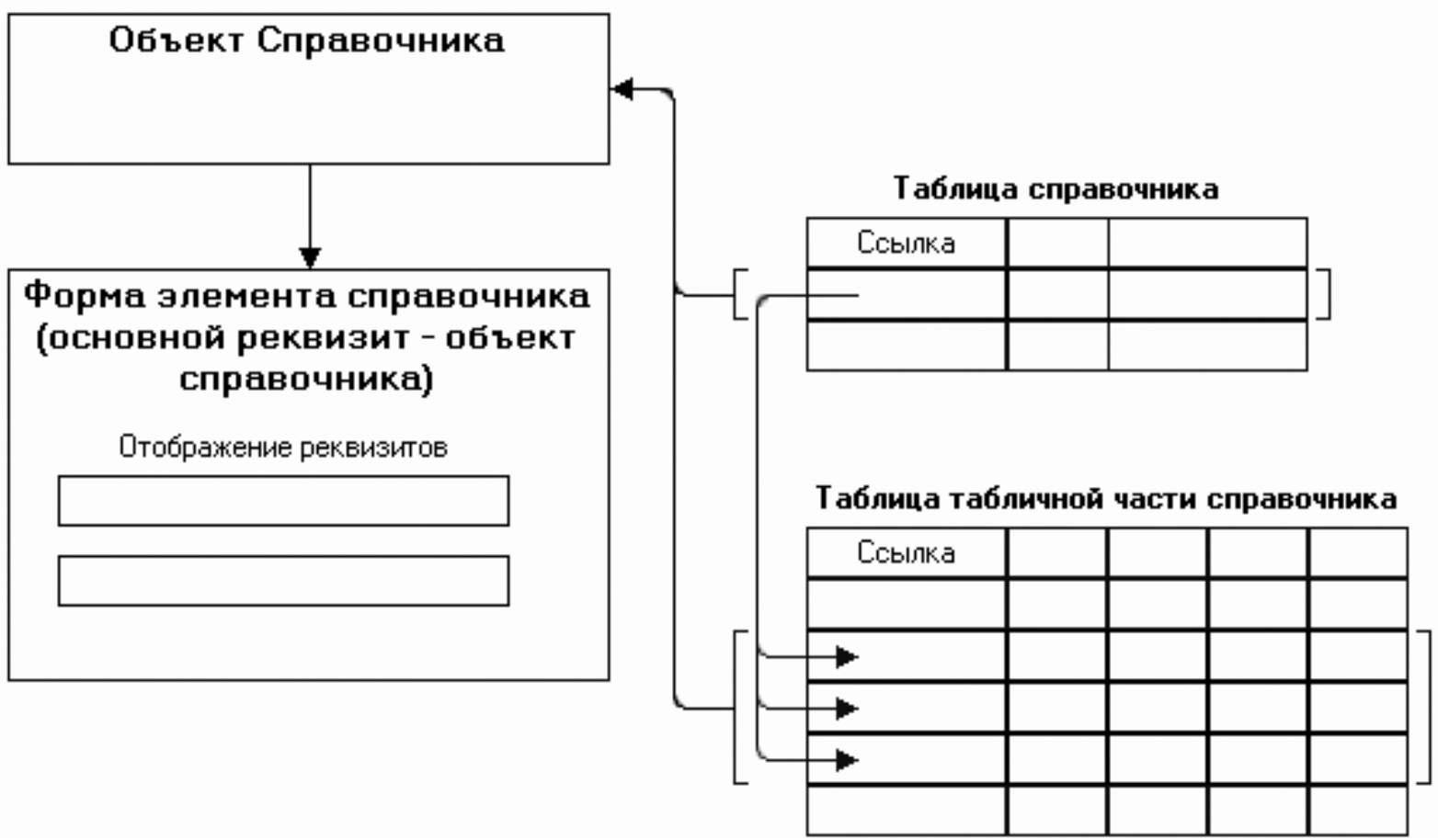
Хранение информации в самих объектах или в других объектах





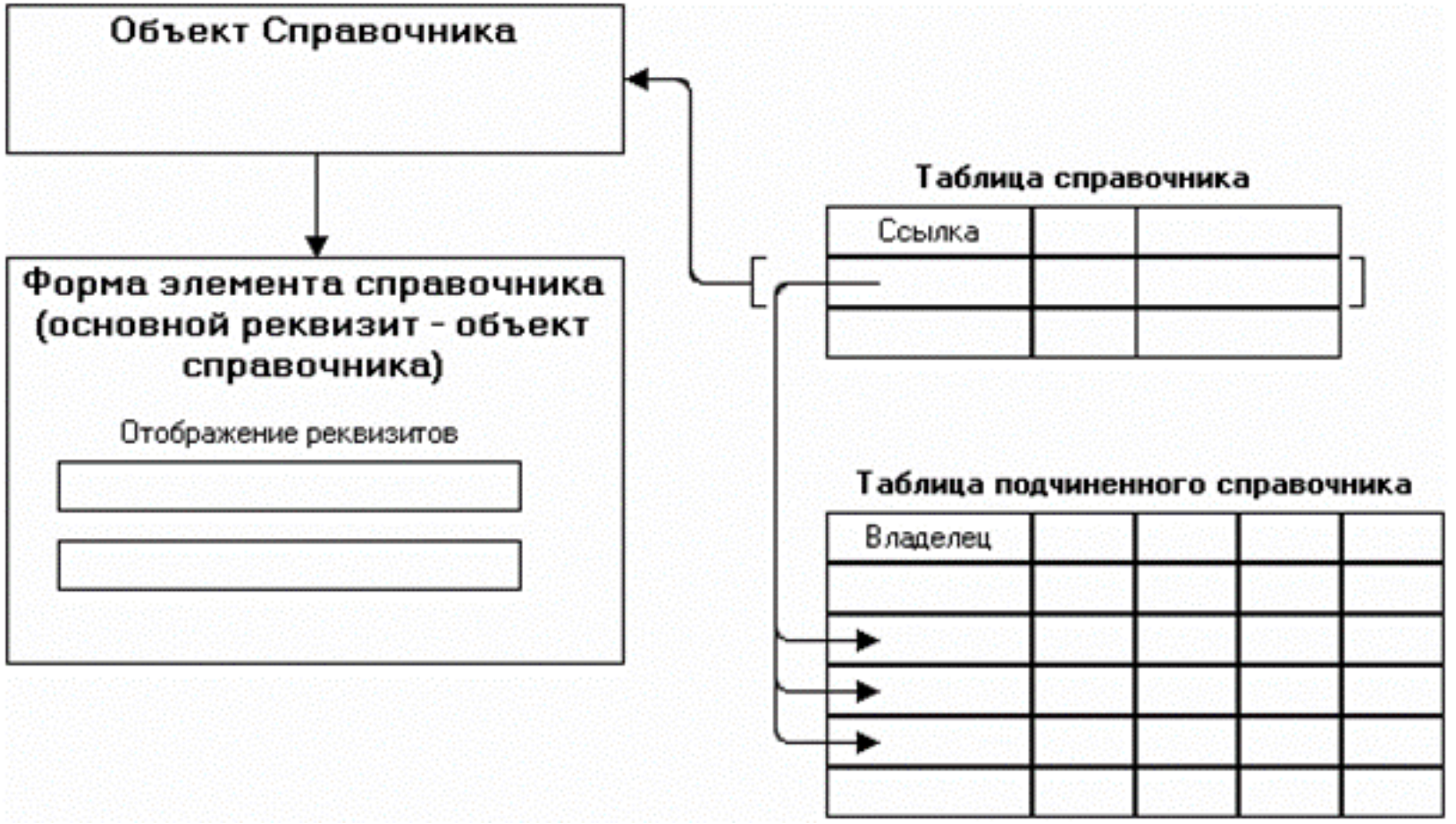
Необходимо помнить, что любой объект – это единая сущность с точки зрения манипулирования данными. То есть при любых операциях с объектом (чтение, запись, модификация) происходит обращение к информации базы данных, касающейся всего объекта.

Объект в любом случае считывается целиком, поскольку при открытии формы платформа также создает объект справочника, обеспечивая тем самым целостность изменений, вносимых в данные объекта как интерактивно, так и программно, в модуле формы



Информацию, которую предполагается хранить в объектах, следует анализировать на предмет разделения на ***активно используемую*** и ***неактивно используемую***.

Например, если информацию спецификаций можно считать «неактивно используемой», ее хранение можно организовать посредством, например, регистра сведений СпецификацииДоговоров или подчиненного справочника СпецификацииДоговоров

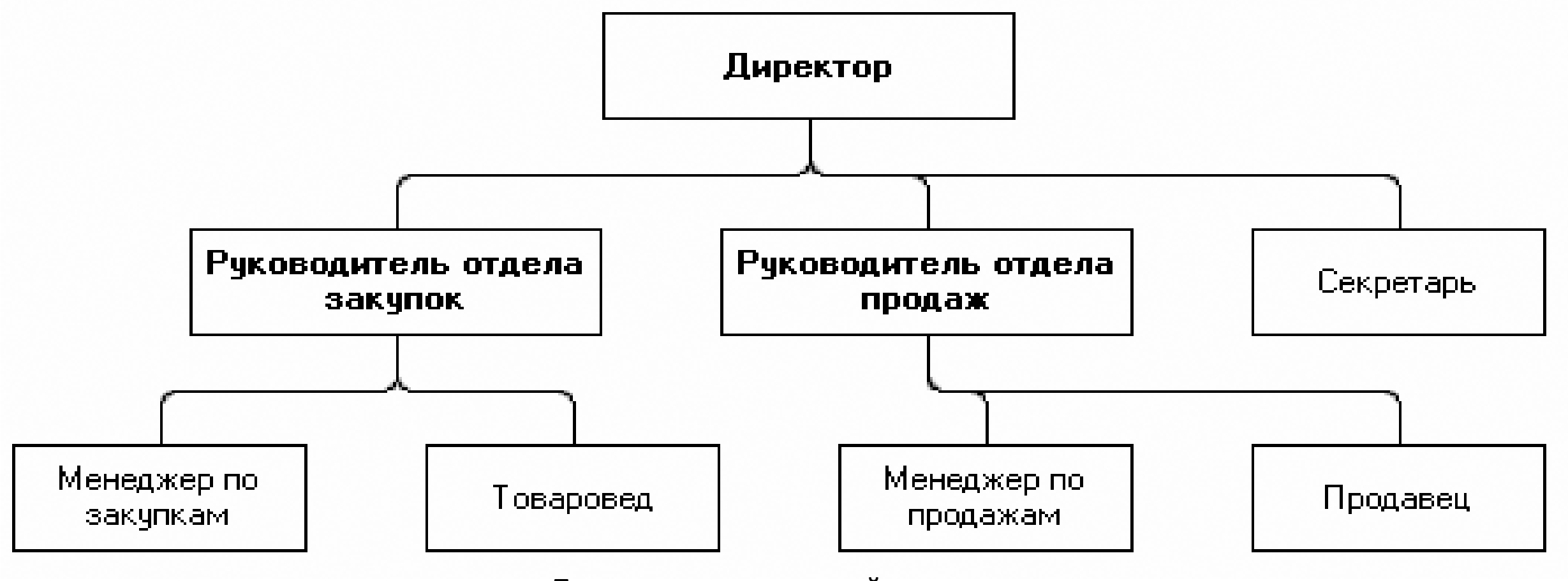


Хранение иерархической информации

Иерархическими принято считать структуры, у которых четко прослеживаются отношения «один ко многим».
Например, «один руководитель – много подчиненных».

Пример одноуровневой иерархии

Хранение иерархической информации



Пример многоуровневой иерархии

Уровней иерархии может быть больше одного. «один руководитель, в подчинении которого находятся руководители подразделений, в подчинении которых находятся подчиненные» – пример трехуровневой иерархической структуры. Ну а в общем случае уровней может быть значительно больше.

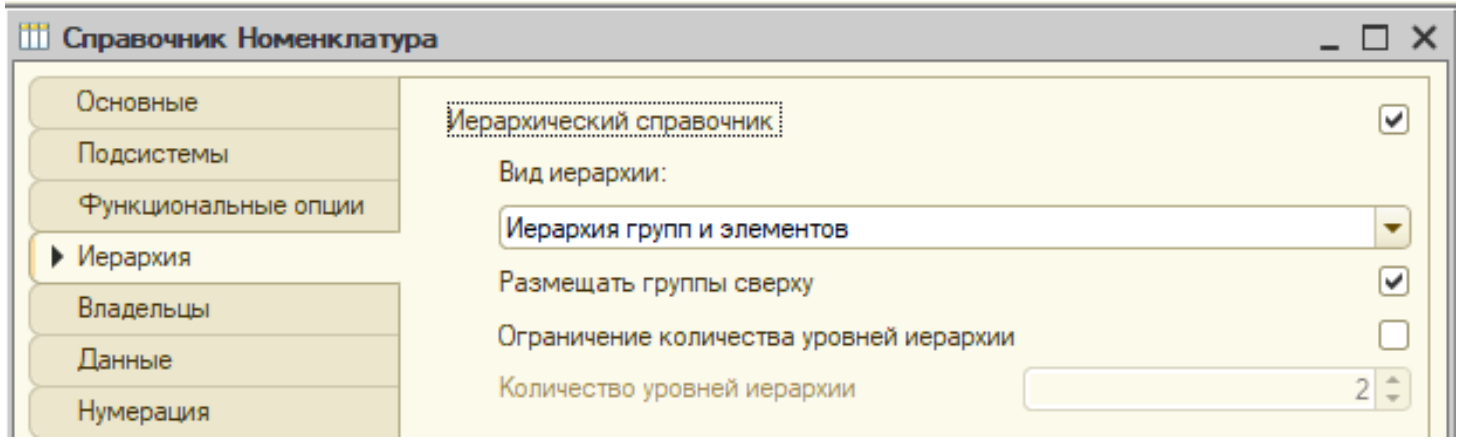
Табличный способ хранения иерархической информации

Сотрудник	Руководитель
Директор	
Руководитель отдела закупок	Директор
Менеджер по закупкам	Руководитель отдела закупок
Товаровед	Руководитель отдела закупок
Руководитель отдела продаж	Директор
Менеджер по продажам	Руководитель отдела продаж
Продавец	Руководитель отдела продаж
Секретарь	Директор

При разработке бизнес-решений разработчику чаще всего приходится иметь дело с иерархическими структурами при организации хранения условно-постоянной информации (справочники, планы видов характеристик), реже – при работе с регистрацией событий. В отношении хранения условно-постоянной информации могут возникать следующие ситуации:

- – иерархия объектных данных одной сущности;
- – хранение иерархии необъектных данных внутри объекта;
- – иерархия объектных данных разных сущностей;
- – хранение иерархии необъектных данных вне объекта.

Хранение иерархии данных одной сущности



Для этого используются иерархические справочники. Чтобы справочник сделать иерархическим, необходимо на закладке Иерархия указать признак Иерархический справочник и выбрать вид иерархии. Дальнейшую работу возьмет на себя платформа. В состав основной таблицы справочника будет добавлено поле Родитель для хранения в записях иерархически подчиненных элементов ссылок на элементы, которым они непосредственно иерархически подчиняются.

Хранение иерархии необъектных данных внутри объекта

Практически все объекты системы «1С:Предприятие», предназначенные для хранения данных объектных сущностей (справочники, документы, планы видов характеристик, планы видов расчета и т. д.), имеют возможность хранить свою информацию не только в виде значений реквизитов, но и в составе ***подчиненных табличных частей***.

Однако необходимо иметь в виду, что информация строк подчиненной табличной части не имеет своей объектной сущности, т.е. нельзя будет создать реквизиты в других объектах, ссылающиеся на строки табличной части. Для задач, когда ссылка на такие сведения смысла не имеет, хранение множества подчиненных данных, характеризующих данный объект, внутри самого объекта может быть весьма удобным. Пример, справочник ПоступлениеТоваров и табличная часть Состав.

Хранение иерархии необъектных данных внутри объекта

Пример иерархической информации

Заказ	Номенклатура	Количество	Цена	Сумма	Пожелания
Заказ № 3 от 12.02.10	Лазерный принтер Canon LBP-810	2	200	400	Белого цвета
					Дополнительно упаковать в гофро-тару
	Телефон LG W7200	10	30	300	Без гарнитуры
					Доставку приурочить к 8 марта
					Цвет любой, только не черный

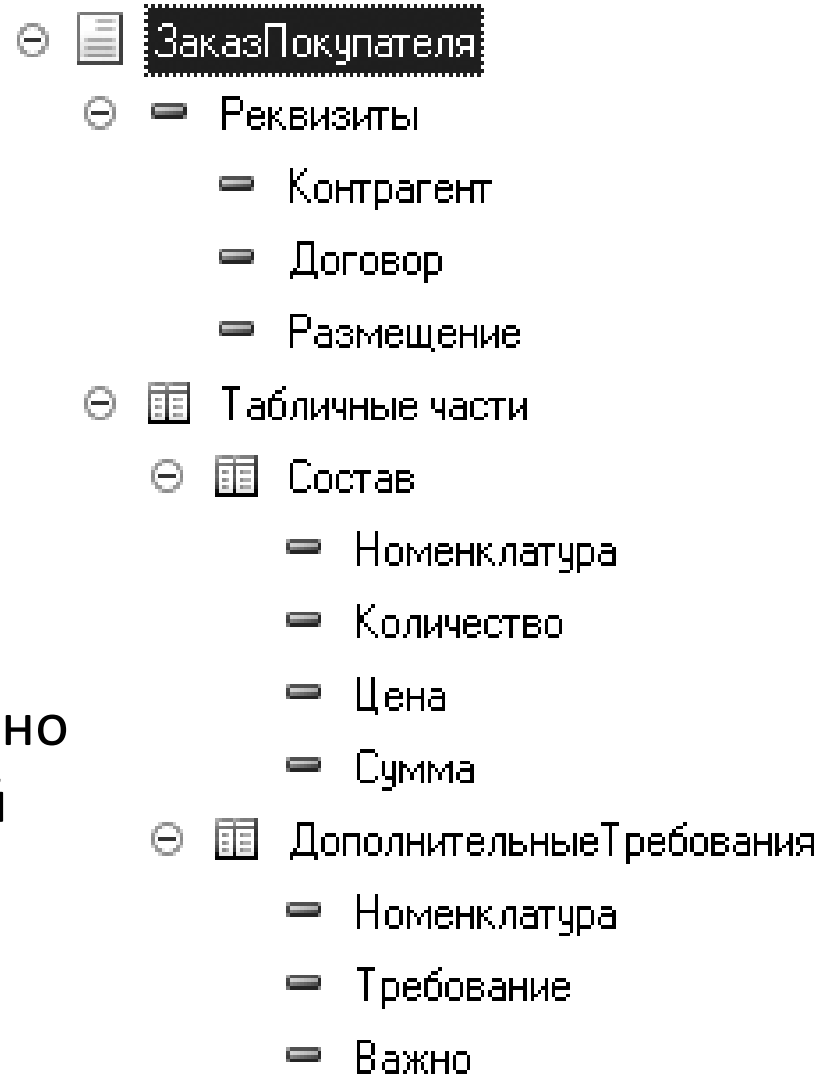
Для хранения информации о дополнительных пожеланиях покупателя, если эту информацию нужно хранить внутри объекта, можно использовать кроме табличной части Состав еще одну табличную часть ДополнительныеТребования.

В основной таблице справочника будет храниться информация в следующих полях:

- *Ссылка,*
- *Код,*
- *Наименование,*
- *Пометка удаления,*
- *Предопределенный,*
- *Родитель,*
- *Владелец,*
- *ЭтоГруппа,*
- *ДатаПоставки,*
- *ВидДоговора.*

Информация табличной части Спецификация – в отдельной таблице, содержащей следующие поля:

- Ссылка (значение этого поля равно значению поля Ссылка основной таблицы справочника);
- НомерСтроки;
- Номенклатура;
- Количество;
- Цена;
- Сумма.



Иерархия объектных данных разных сущностей.

Используются ***подчиненные справочники***. Пример: справочник Сотрудник и подчиненный справочник ТрудоваяДеятельность.

Подчиненные табличные части предназначены для учета только ***необъектных сущностей***, то есть при таком решении в других объектах системы невозможно будет использовать поля или реквизиты, соответствующие понятию «место предыдущей работы конкретного сотрудника».

Если же необходимость в такой объектной сущности есть или может появиться, то решение видится в использовании подчиненного справочника ТрудоваяДеятельность. В его элементах можно хранить информацию о должности, месте работы, дате начала деятельности, дате окончания деятельности по каждому эпизоду предыдущей работы сотрудника.

Хранение иерархии необъектных данных вне объекта

Для хранения необъектных данных, соответствующих определенным комбинациям значений измерений, предназначены ***регистры сведений.***

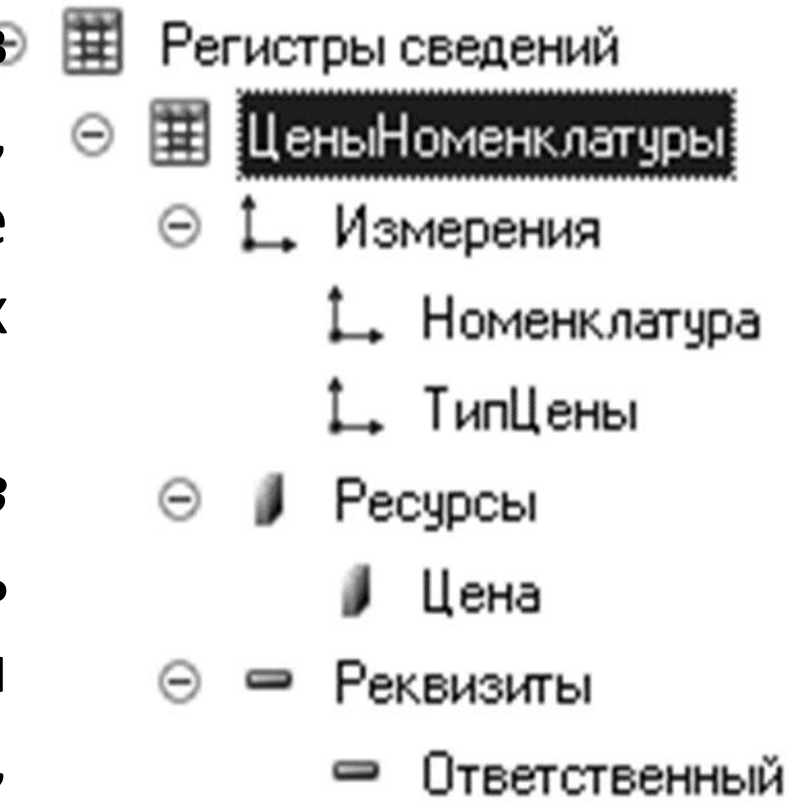
Регистры сведений позволяют хранить не только статические варианты иерархических структур, но и изменяющиеся во времени. Если указанные регистры сведений сделать периодическими, то впоследствии можно будет легко хранить динамику картины отношений.

Также для ситуаций, когда в готовое (и эксплуатируемое) решение приходится вводить новые сущности, добавление новых измерений к регистру сведений позволяет легко решать задачи усложнения «взаимоотношений» данных.

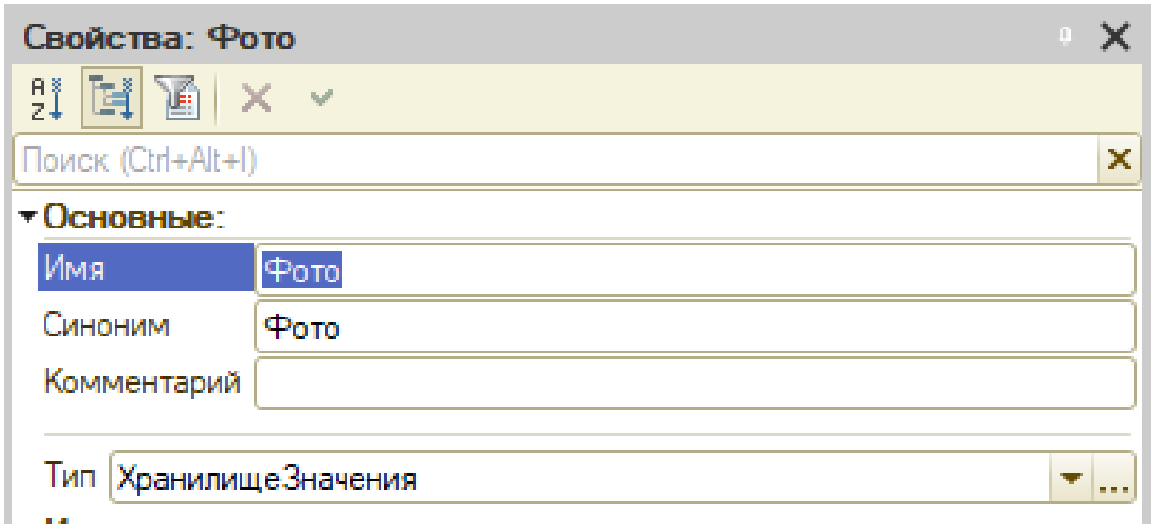
Хранение информации, имеющей привязку ко времени

Для решения учетных и аналитических задач кроме учета событий важно, чтобы система в определенных ситуациях учитывала показатели, которые изменяются или при обработке вышеуказанных событий, или вследствие других действий пользователя.

Использование ***периодических регистров сведений*** позволяет достаточно быстро создать решение, большую часть функциональности которого поддерживает сама платформа, например, получение информации об актуальных значениях цен на некоторый момент времени



Хранение вспомогательной информации в объекте «ХранилищеЗначений»



Особенностью использования таких полей является то, что база данных «не обязана ничего знать» о природе хранимой там информации. Это могут быть картинки, файлы, таблицы, данные примитивных типов и т.п.

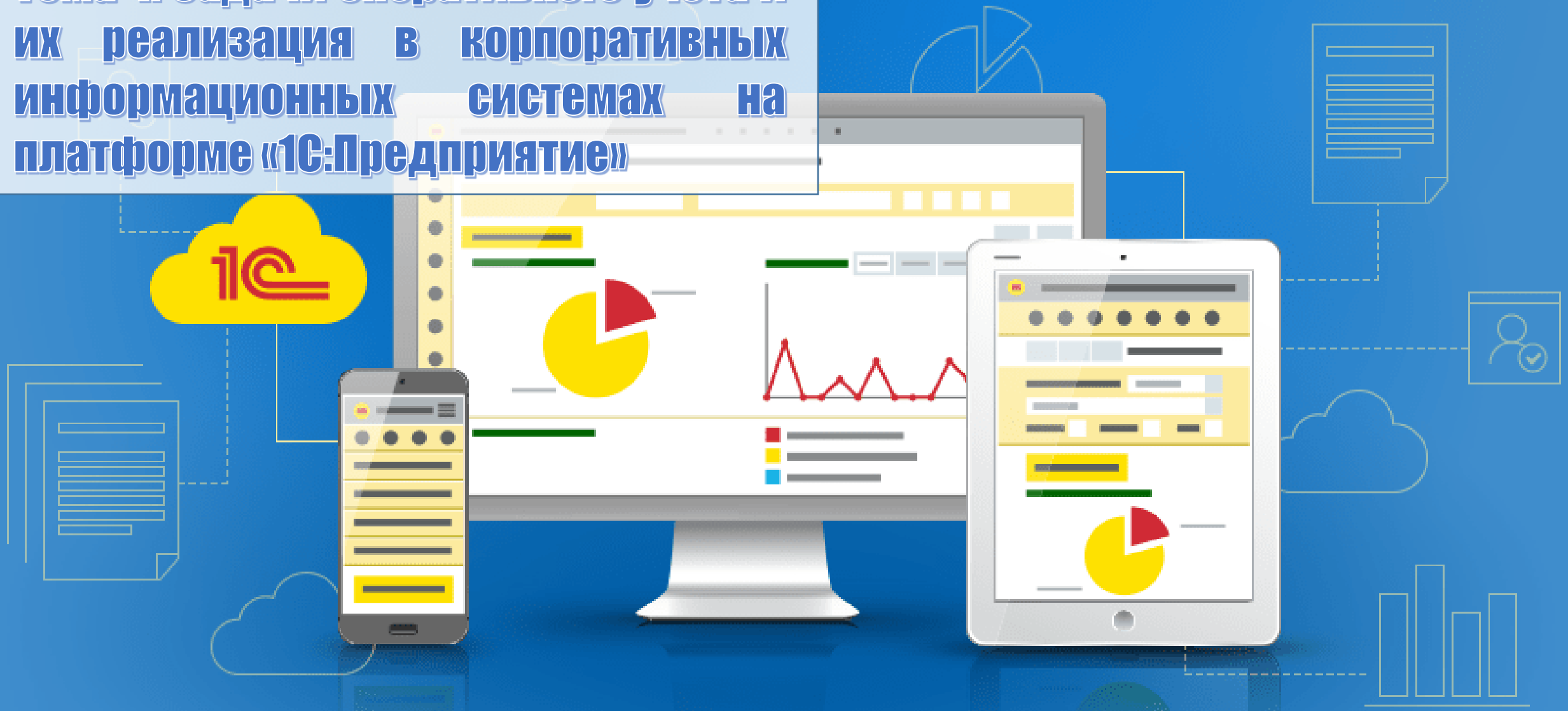
База данных отвечает лишь за хранение этой информации, не обеспечивая больше никакой функциональности. При этом еще говорят, что тип значения ХранилищеЗначения позволяет хранить значения различных типов в сериализованном виде, то есть в виде, который позволяет записывать данные и восстанавливать их. Важной особенностью хранилища значений является возможность хранения данных в сжатом виде.

С прикладной точки зрения в полях типа ХранилищеЗначения можно хранить, например, фотографии сотрудников и т.п.

Особенности Хранение вспомогательной информации в объекте «ХранилищеЗначений»

- Следует осмотрительно подходить к решению, что именно требуется хранить в базе данных. Большой объем хранимой вспомогательной информации может привести к тому, что административные операции (например, создание резервной копии базы данных) выполняются медленно из-за большого объема базы данных.
- Особенно аккуратно нужно относиться к возможности использования полей типа ХранилищеЗначения в составе объектов (например, справочников, документов), активно использующихся при реализации основной бизнес-логики. Ведь данные объектов считываются целиком при обращении к ним. Поэтому, например, вопрос хранения тех же фотографий лучше решать не в составе справочника Сотрудники, а в отдельном подчиненном справочнике или регистре сведений.

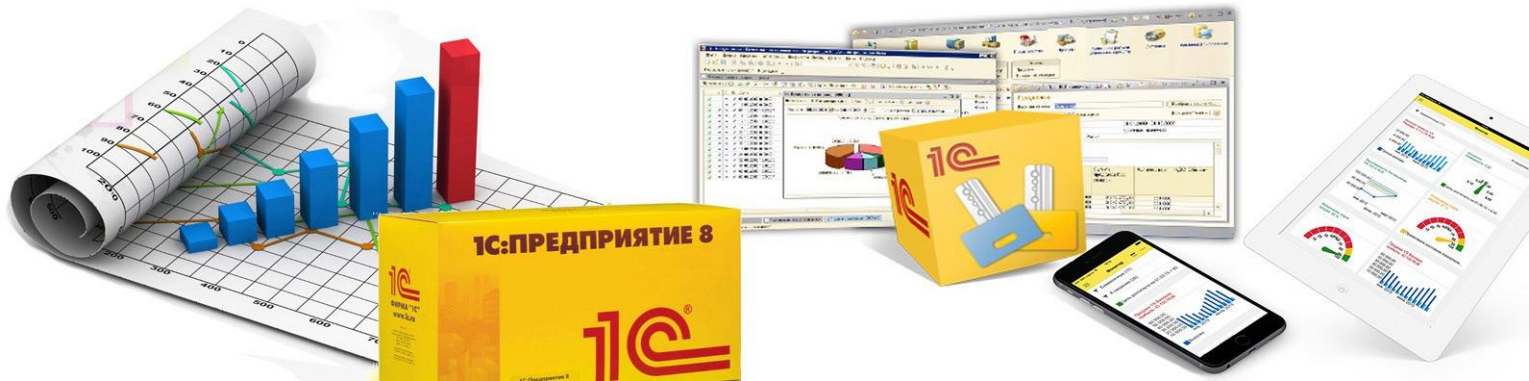
Тема 4. Задачи оперативного учета и их реализация в корпоративных информационных системах на платформе «1С:Предприятие»



- *Задачи оперативного учета в корпоративных информационных системах.*
- *Виды регистров накопления: регистры остатков и регистры оборотов.*
- *Структура регистра накопления: измерения, ресурсы, реквизиты.*
- *Регистры сведений: структура регистра сведений.*
- *Периодические регистры сведений.*
- *Подчинение записей регистратору.*
- *Проектирование структуры регистров сведений.*

Оперативный учет: задачи, решаемые регистрами накопления

Оперативный учет – учет, позволяющий максимально быстро получать информацию о значениях показателей, учитываемых в автоматизируемой системе. При автоматизации учета движения средств чаще приходится сталкиваться с ситуацией, когда в момент регистрации изменений показателя фиксируется не конечное итоговое значение показателя, а его приращение. А вот при получении данных из системы учета уже требуются накопленные (итоговые) значения показателей. Такие показатели называют ***показателями накопления.***

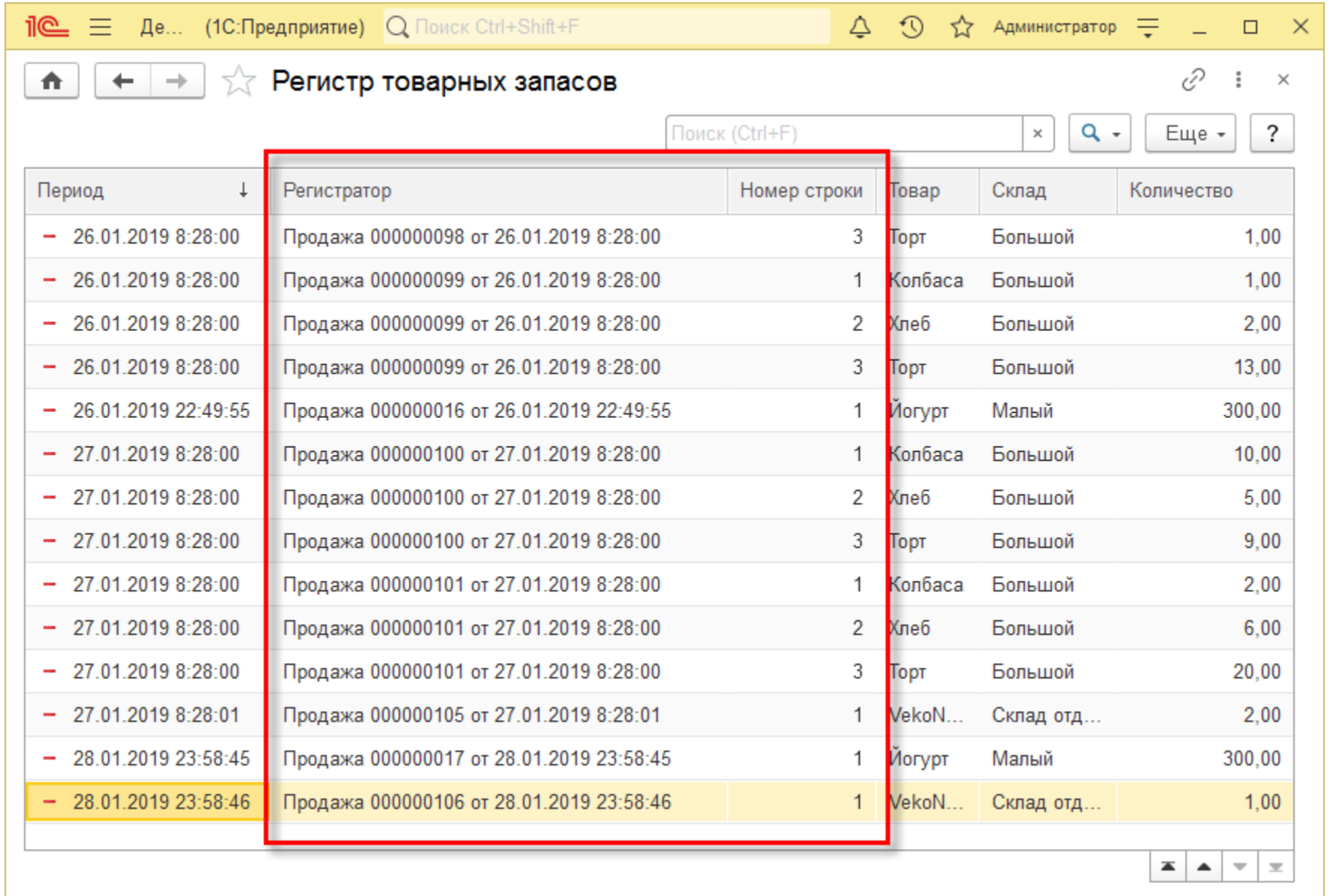


Регистр накопления

Регистры накопления — это прикладные объекты конфигурации. Они составляют основу механизма учета движения средств (финансов, товаров, материалов и т. д.), который позволяет автоматизировать такие направления, как складской учет, взаиморасчеты, планирование.

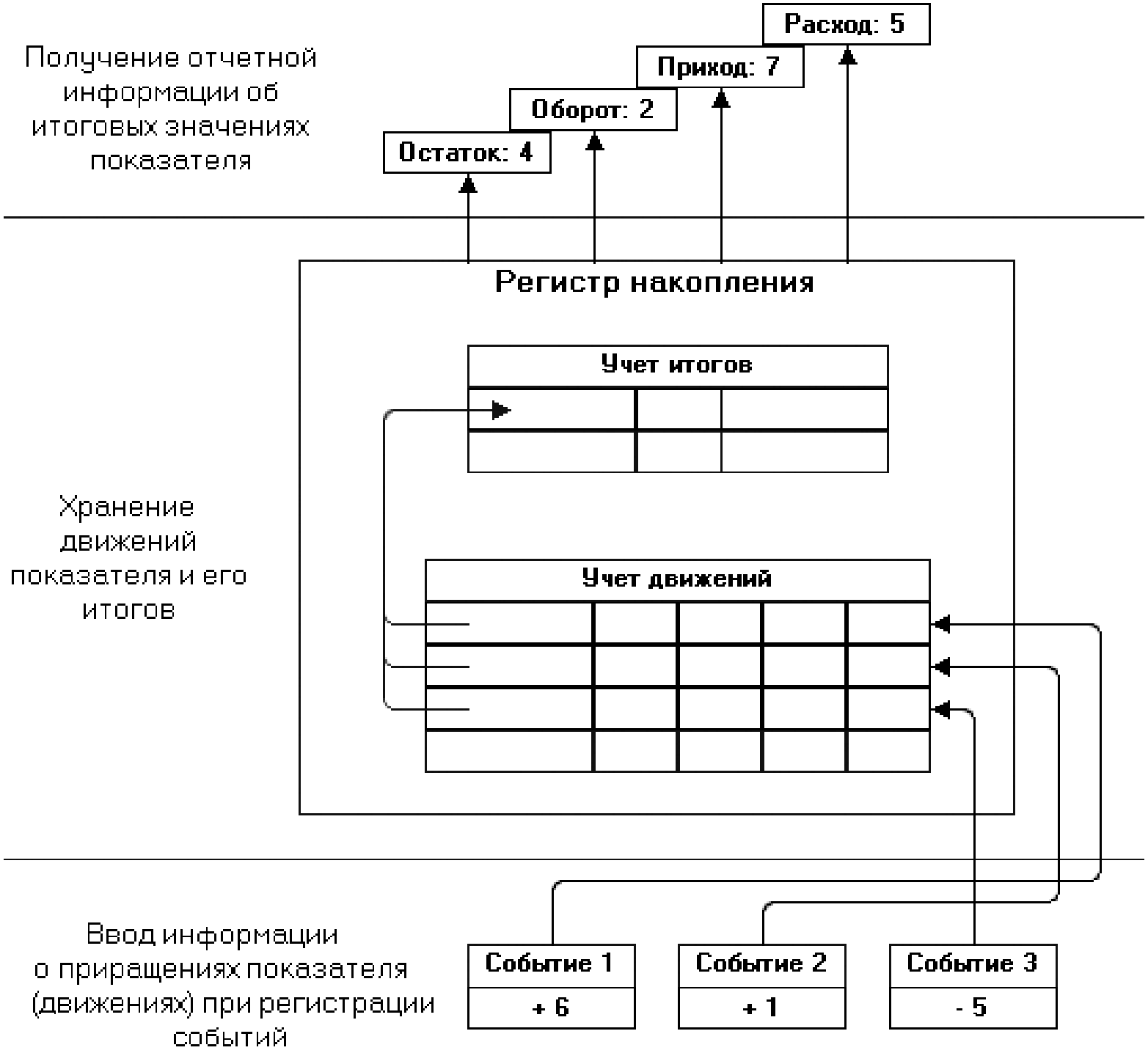
Обоснованность регистрации изменений показателей

Информация о приращениях показателей (а они могут быть как положительными, так и отрицательными) вносится в регистр накопления только посредством движений, то есть посредством наборов записей регистра, подчиненных ***документу-регистратору***. Этим обеспечивается ***обоснованность*** регистрации изменений показателей – регистрацией событий, приводящих к этим изменениям.



Период	Регистратор	Номер строки	Товар	Склад	Количество
26.01.2019 8:28:00	Продажа 000000098 от 26.01.2019 8:28:00	3	Торт	Большой	1,00
26.01.2019 8:28:00	Продажа 000000099 от 26.01.2019 8:28:00	1	Колбаса	Большой	1,00
26.01.2019 8:28:00	Продажа 000000099 от 26.01.2019 8:28:00	2	Хлеб	Большой	2,00
26.01.2019 8:28:00	Продажа 000000099 от 26.01.2019 8:28:00	3	Торт	Большой	13,00
26.01.2019 22:49:55	Продажа 000000016 от 26.01.2019 22:49:55	1	Йогурт	Малый	300,00
27.01.2019 8:28:00	Продажа 000000100 от 27.01.2019 8:28:00	1	Колбаса	Большой	10,00
27.01.2019 8:28:00	Продажа 000000100 от 27.01.2019 8:28:00	2	Хлеб	Большой	5,00
27.01.2019 8:28:00	Продажа 000000100 от 27.01.2019 8:28:00	3	Торт	Большой	9,00
27.01.2019 8:28:00	Продажа 000000101 от 27.01.2019 8:28:00	1	Колбаса	Большой	2,00
27.01.2019 8:28:00	Продажа 000000101 от 27.01.2019 8:28:00	2	Хлеб	Большой	6,00
27.01.2019 8:28:00	Продажа 000000101 от 27.01.2019 8:28:00	3	Торт	Большой	20,00
27.01.2019 8:28:01	Продажа 000000105 от 27.01.2019 8:28:01	1	VekoN...	Склад отд...	2,00
28.01.2019 23:58:45	Продажа 000000017 от 28.01.2019 23:58:45	1	Йогурт	Малый	300,00
28.01.2019 23:58:46	Продажа 000000106 от 28.01.2019 23:58:46	1	VekoN...	Склад отд...	1,00

Упрощенная схема регистра накопления



Виды накапливаемых показателей

Показатели накопления имеют различный прикладной смысл. Различают следующие виды накапливаемых показателей:

– показатели остатков,

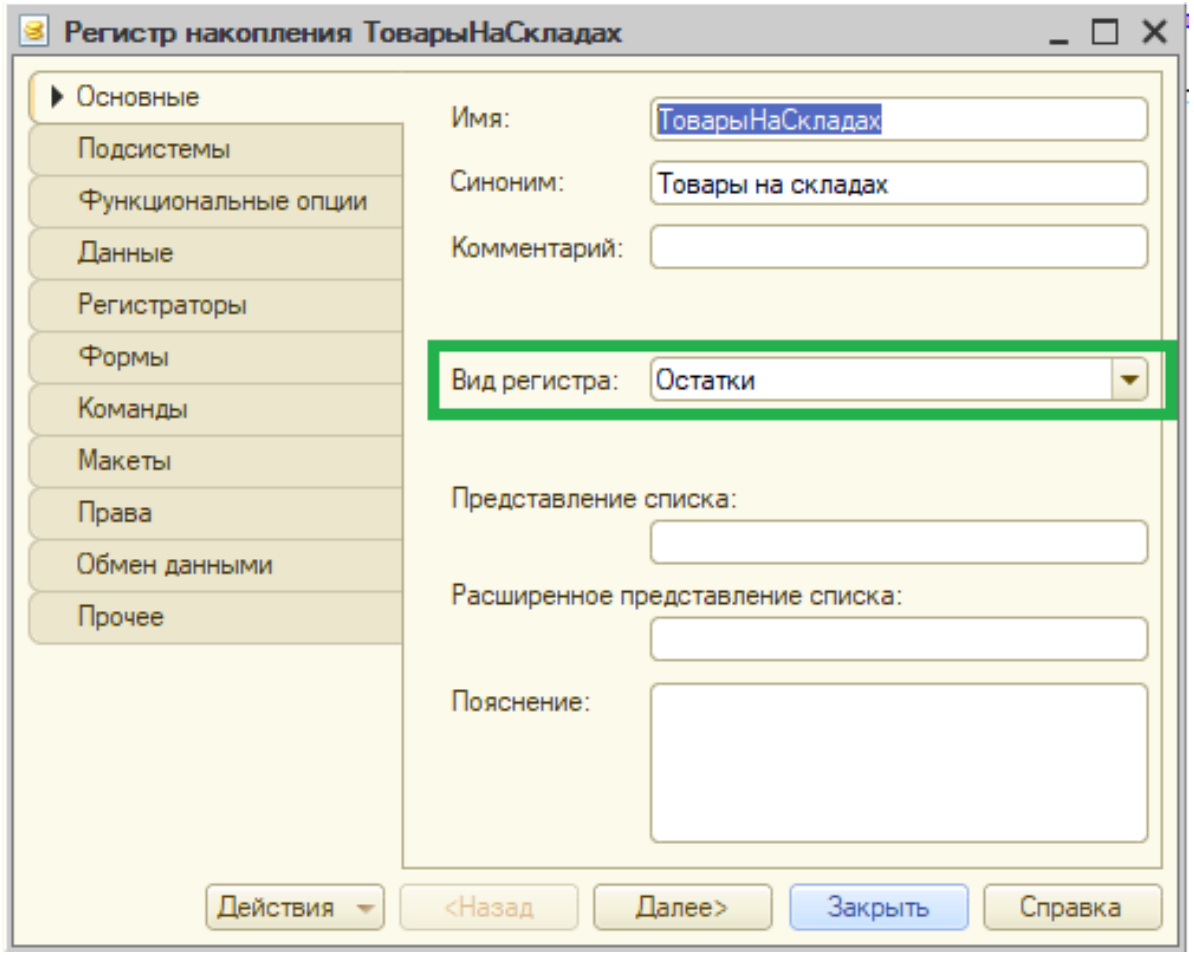
– оборотные показатели.

Упрощенно можно сравнить регистр накопления с неким «черным ящиком», «на вход» которого подаются значения приращений, а «на выходе» можно получить накопленные значения приращений

Для гибкого и эффективного решения задач прикладной области система «1С:Предприятие» позволяет использовать два вида регистров накопления. Виды регистров накопления: ***регистры остатков*** и ***регистры оборотов***.

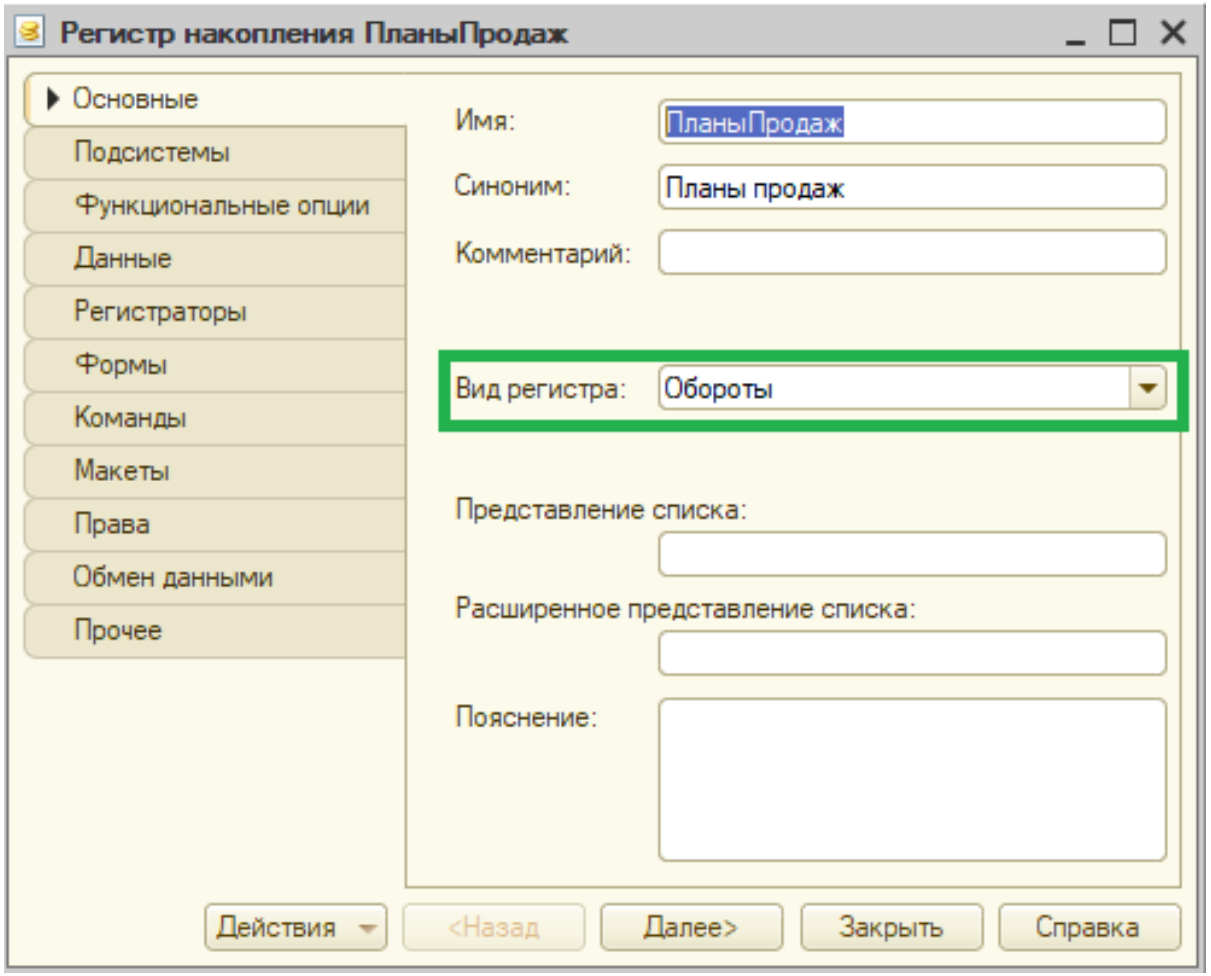
Регистры накопления остатков

Позволяют получать итоговые значения показателей остатков и, кроме того (суммируя приращения этих показателей за периоды), позволяют получать обороты. Например, при решении задачи учета товаров на складах может понадобиться как значение остатка товаров на момент времени, так и оборот поступлений или расходов товара за периоды времени.



Оборотные регистры накопления

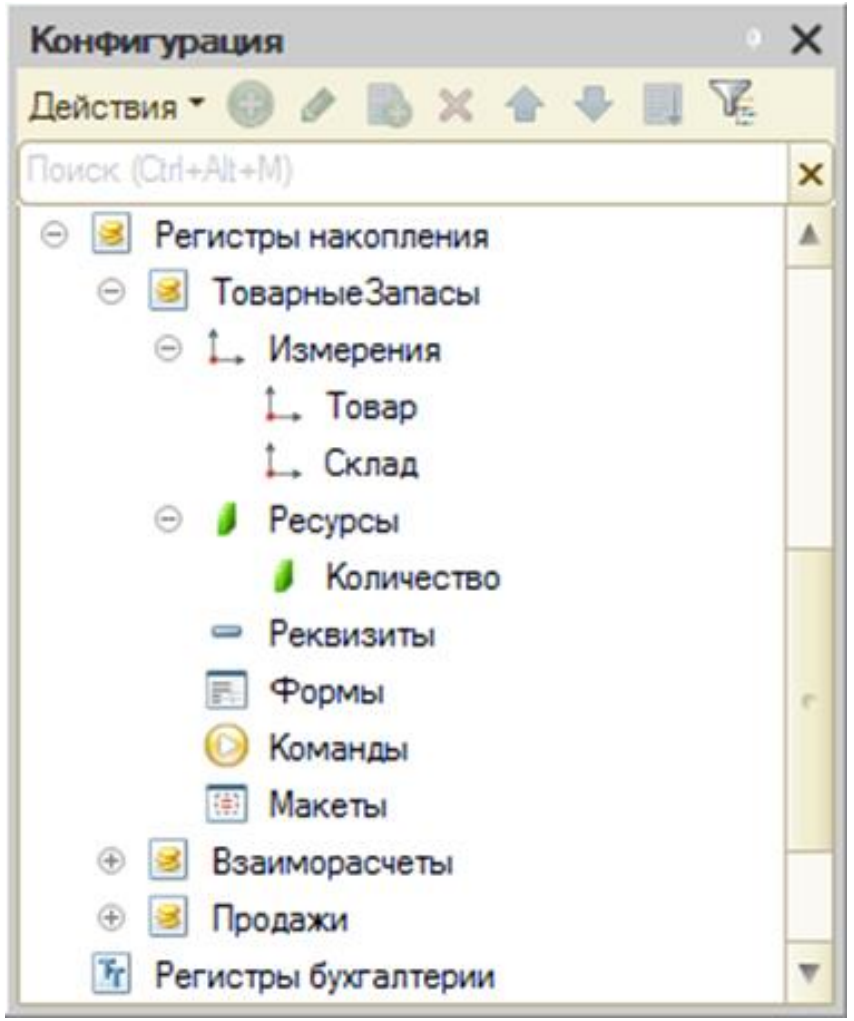
Если для некоторых сущностей накопление остатков смысла не имеет и требуется накапливать только обороты, тогда следует использовать оборотные регистры накопления.



Структура регистра накопления

В состав регистра накопления как объекта конфигурации входят:

- измерения,
- ресурсы,
- реквизиты.

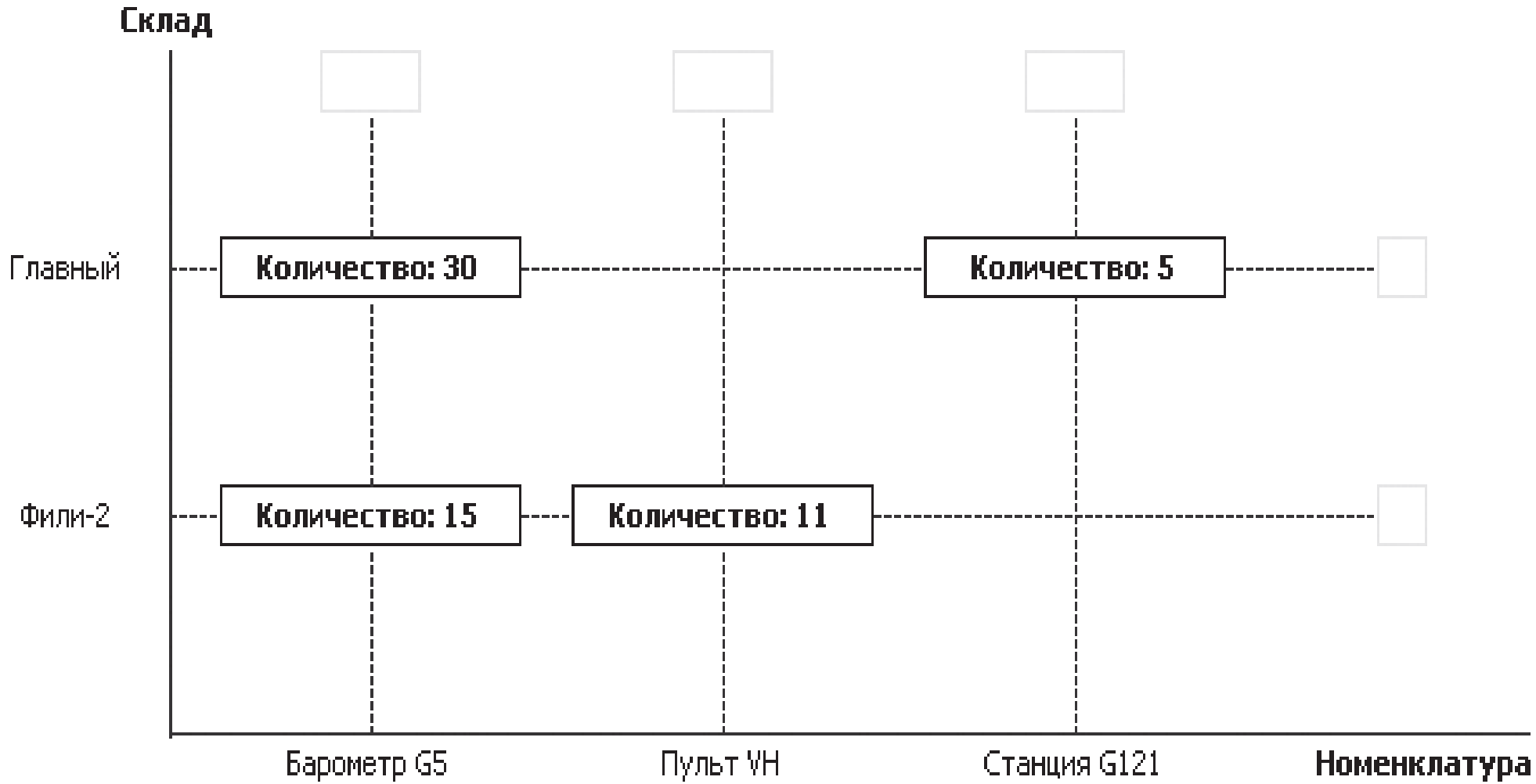


В регистрах накопления разрезы учета реализуются с помощью ***измерений***.

Ресурсы используются для хранения информации как о приращениях, так и о самих значениях показателей. По сути, каждый ресурс хранит данные одного показателя.

Реквизиты – дополнительные характеристики движений, то есть первичных записей регистра. Реквизиты не влияют на итоги, хранимые в регистре.

В общем случае регистр накопления можно представить как *n*-мерную систему разрезов учета (измерений), в узлах которой хранятся совокупные данные ресурсов (итоги).



Данные каждого регистра накопления хранятся в базе данных в двух таблицах:
– ***таблица движений регистра накопления***,
– ***таблица итогов регистра накопления***.

Структура таблиц базы данных для регистров накопления разных видов тоже отличается. Она оптимизирована под решение соответствующих задач.

Состав колонок таблицы движений регистра накопления остатков

- ***Период*** – дата записи. Совместно с полями Регистратор и НомерСтроки определяет положение данной записи на временной оси;
- ***Регистратор*** – ссылка на документ, которому подчинена данная запись;
- ***НомерСтроки*** – уникальный номер данной записи в наборе записей регистра, подчиненных документу, указанному в поле Регистратор;
- ***ВидДвижения*** – значение системного перечисления ВидДвиженияНакопления, обозначающее направление приращения указанных в записи ресурсов на итоги по этим ресурсам (Приход или Расход);
- ***Активность*** – тип Булево. Содержит признак влияния записи на итоги регистра;
- ***<Измерение>*** – значение измерения. Количество таких полей равно количеству измерений, определенных в данных регистра как объекта конфигурации;
- ***<Ресурс>*** – значение ресурса. Количество таких полей равно количеству ресурсов, определенных в данных регистра как объекта конфигурации;
- ***<Реквизит>*** – значение реквизита. Количество таких полей равно количеству реквизитов, определенных в данных регистра как объекта конфигурации.

Состав колонок таблицы итогов регистра накопления остатков

- ***Период*** – дата, на которую актуально состояние хранимого в таблице итога;
- ***<Измерение>*** – значение измерения – разреза учета хранимых итогов. Количество таких полей равно количеству измерений, определенных в данных регистра как объекта конфигурации;
- ***<Ресурс>*** – значение итога ресурса. Количество таких полей равно количеству ресурсов, определенных в данных регистра как объекта конфигурации;
- ***Разделитель*** – поле, позволяющее распараллелить обновление записей итогов. Добавляется в структуру таблицы итогов для регистров накопления, у которых установлено свойство Разрешить разделение итогов.

Состав колонок таблицы движений регистра накопления оборотов

- ***Период*** – дата записи. Совместно с полями Регистратор и НомерСтроки определяет положение данной записи на временной оси;
- ***Регистратор*** – ссылка на документ, которому подчинена данная запись;
- ***НомерСтроки*** – уникальный номер данной записи в наборе записей регистра, подчиненных документу, указанному в поле Регистратор;
- ***Активность*** – тип Булево. Содержит признак влияния записи на итоги регистра;
- ***<Измерение>*** – значение измерения. Количество таких полей равно количеству измерений, определенных в данных регистра как объекта конфигурации;
- ***<Ресурс>*** – значение ресурса. Количество таких полей равно количеству ресурсов, определенных в данных регистра как объекта конфигурации;
- ***<Реквизит>*** – значение реквизита. Количество таких полей равно количеству реквизитов, определенных в данных регистра как объекта конфигурации.

Структура таблиц оборотного регистра накопления, хранимых в базе данных, схожа с аналогичными таблицами для регистра накопления остатков. Отличие заключается только в том, что для оборотных регистров не существует понятия вид движения и понятия текущие итоги.

Состав колонок таблицы итогов регистра накопления оборотов

- ***Период*** – дата, на которую актуально состояние хранимого в таблице итога;
- ***<Измерение>*** – значение измерения – разреза учета хранимых итогов. Количество таких полей равно количеству измерений, определенных в данных регистра как объекта конфигурации;
- ***<Ресурс>*** – значение итога ресурса. Количество таких полей равно количеству ресурсов, определенных в данных регистра как объекта конфигурации;
- ***Разделитель*** – поле, позволяющее распараллелить обновление записей итогов. Добавляется в структуру таблицы итогов для регистров накопления, у которых установлено свойство Разрешить разделение итогов.

Как видно, структура таблицы итогов оборотного регистра накопления тоже схожа

Обоснованность информации регистров

Регистры накопления не поддерживают независимого формирования записей без использования ***документа-регистратора***. Этим достигается ***обоснованность информации регистров*** – данными документов. Т.е. обоснованность информации объектов, осуществляющих учет показателей, данным объектов, осуществляющих первичную регистрацию событий, приводящих к изменению значений показателей. С другой стороны, к каждому регистратору может быть отнесено более одной записи движения. Поэтому для регистров накопления в таблице движений ключевыми являются поля Регистратор и НомерСтроки.

Манипулирование записями регистра

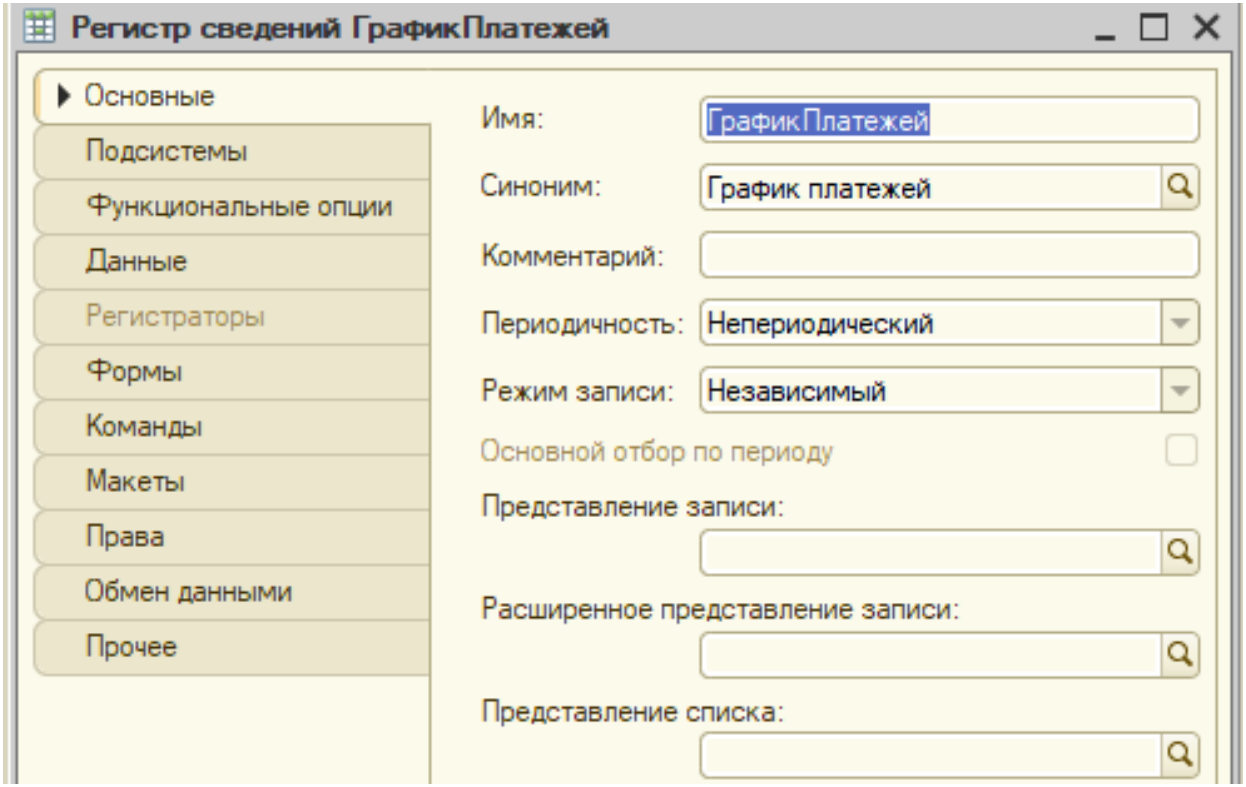
- Для обеспечения манипулирования записями регистра при формировании или модифицировании движений используется объект встроенного языка системы РегистрНакопленияНаборЗаписей.<имя>.
- Эти действия выполняются при использовании обязательного ***отбора по регистратору.***
- для заполнения или модификации данных таблицы движений есть две возможности:
  - посредством создания набора записей регистра с обязательным отбором по регистратору;
  - посредством использования свойства объекта документа Движения, представляющего собой коллекцию наборов записей, подчиненных данному регистратору.

Структура регистра сведений

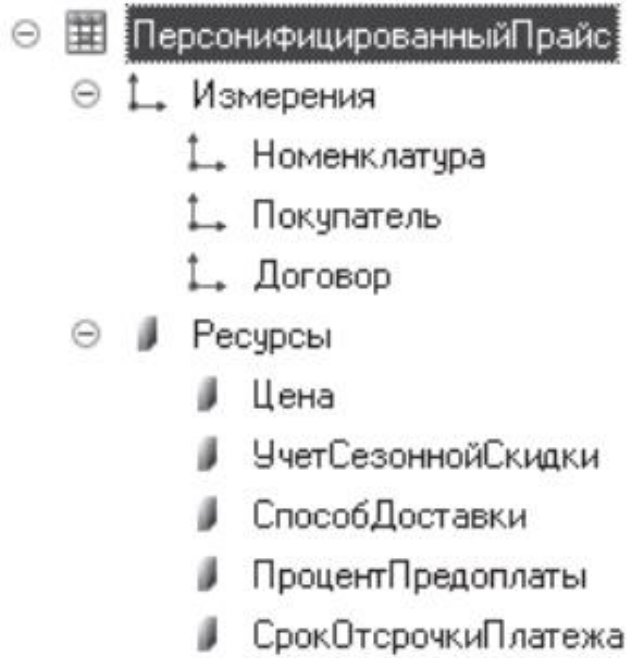
Объект 1С «Регистры сведений» – это прикладные объекты конфигурации, которые позволяют хранить в прикладном решении произвольные данные в разрезе нескольких измерений.

Для решения таких задач регистры сведений обладают функциональностью, которая включает в себя:

- ❑ уникальность записей в разрезах ключей записей,
- ❑ периодичность,
- ❑ подчинение записей регистратору.



- Регистр сведений фактически представляет собой в общем случае многомерный массив данных, необходимый для реализации функции, которая может выдать нужную информацию по определенному набору аргументов. Аргументы функции называются измерениями, а результат функции – ресурсами. Количество используемых в составе регистра сведений измерений и ресурсов определяется задачами, которые решаются посредством этого регистра.
- Внесение изменений в регистр сведений может выполняться: вручную; при помощи документов. На этапе проектирования разработчик для этого определяет режим записи в регистр сведений: Независимый или Подчинение регистратору.



Периодические регистры сведений

Одной из возможностей регистра сведений является хранение данных не только в разрезе указанных измерений, но и в разрезе времени. Главное отличие периодического регистра сведений от обычного заключается в том, что в нем присутствует дополнительное системное измерение "Период", имеющее тип "дата". Это позволяет получать не только текущие сведения об объекте, но также на любой момент времени.

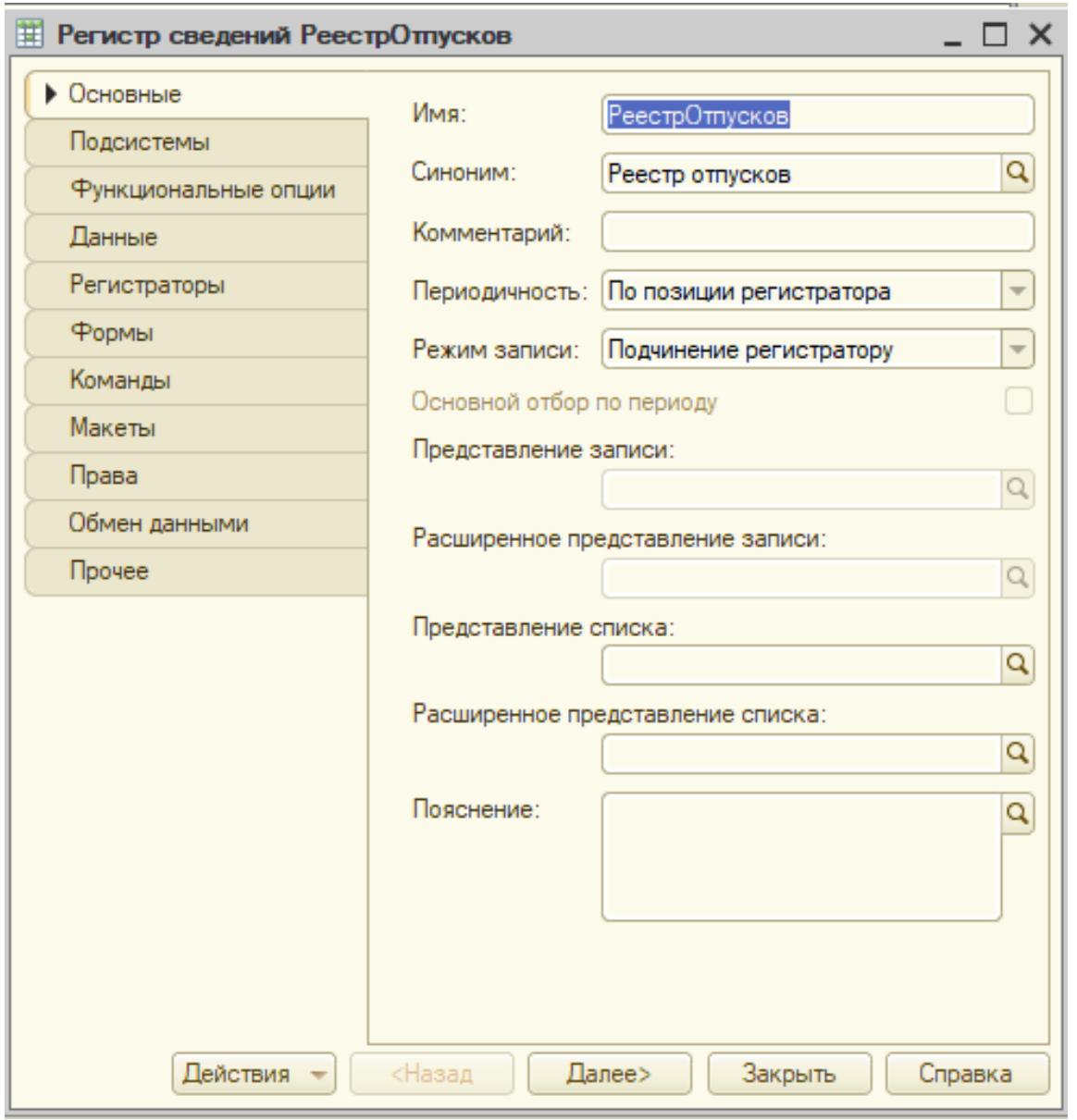
Данное свойство позволяет добавить к списку измерений регистра дополнительное измерение — «Период». Разработчик может указать минимальную периодичность, с которой записи будут заноситься в регистр.

Периодичность может принимать следующие значения:

- «Непериодический»;
- «В пределах секунды»;
- «В пределах дня»;
- «В пределах месяца»;
- «В пределах квартала»;
- «В пределах года».

Подчинение записей регистратору

Использование значения свойства ***Режим записи*** равным *Подчинение регистратору* может потребоваться в случае, когда логика работы прикладного решения требует того, чтобы изменения, выполняемые в регистре сведений, были жестко связаны с документами, фиксирующими факты хозяйственной деятельности. Такая настройка приводит к включению в состав регистра полей ***Регистратор***, ***НомерСтроки*** и ***Активность***.



Проектирование структуры регистров сведений

В состав регистра сведений как объекта конфигурации включаются: измерения, ресурсы, реквизиты.

Данные каждого регистра сведений хранятся в отдельной таблице базы данных.

Состав колонок регистра сведений

Колонка	Описание
1	2
Период	дата записи, которая определяет положение данной записи на временной оси. Это поле существует только для периодических регистров
Регистратор	содержит ссылку на документ, которому подчинена данная запись. Это поле существует только для регистров с режимом записи *Подчинение регистратору*
НомерСтроки	уникальный номер данной записи в наборе записей регистра, подчиненных документу, указанному в поле *Регистратор*. Это поле существует только для регистров с режимом записи *Подчинение регистратору*
Активность	тип *Булево*. Содержит признак влияния записи на получение информации из регистра. Записи, для которых значение данного свойства установлено в *Ложь*, не будут учитываться при получении «первых» или «последних» записей регистра, а также при получении сведений на определенный момент времени. Существует только для регистров с режимом записи Подчинение регистратору
Ключ	короткий ключ записи регистра. Поле присутствует у непериодических регистров, имеющих хотя бы одно измерение
<Измерение>	содержит значение измерения. Количество таких полей равно количеству измерений, определенных для регистра как объекта конфигурации
<Ресурс>	значение ресурса. Количество таких полей равно количеству ресурсов, определенных для регистра как объекта конфигурации
<Реквизит>	значение реквизита. Количество таких полей равно количеству реквизитов, определенных для регистра как объекта конфигурации

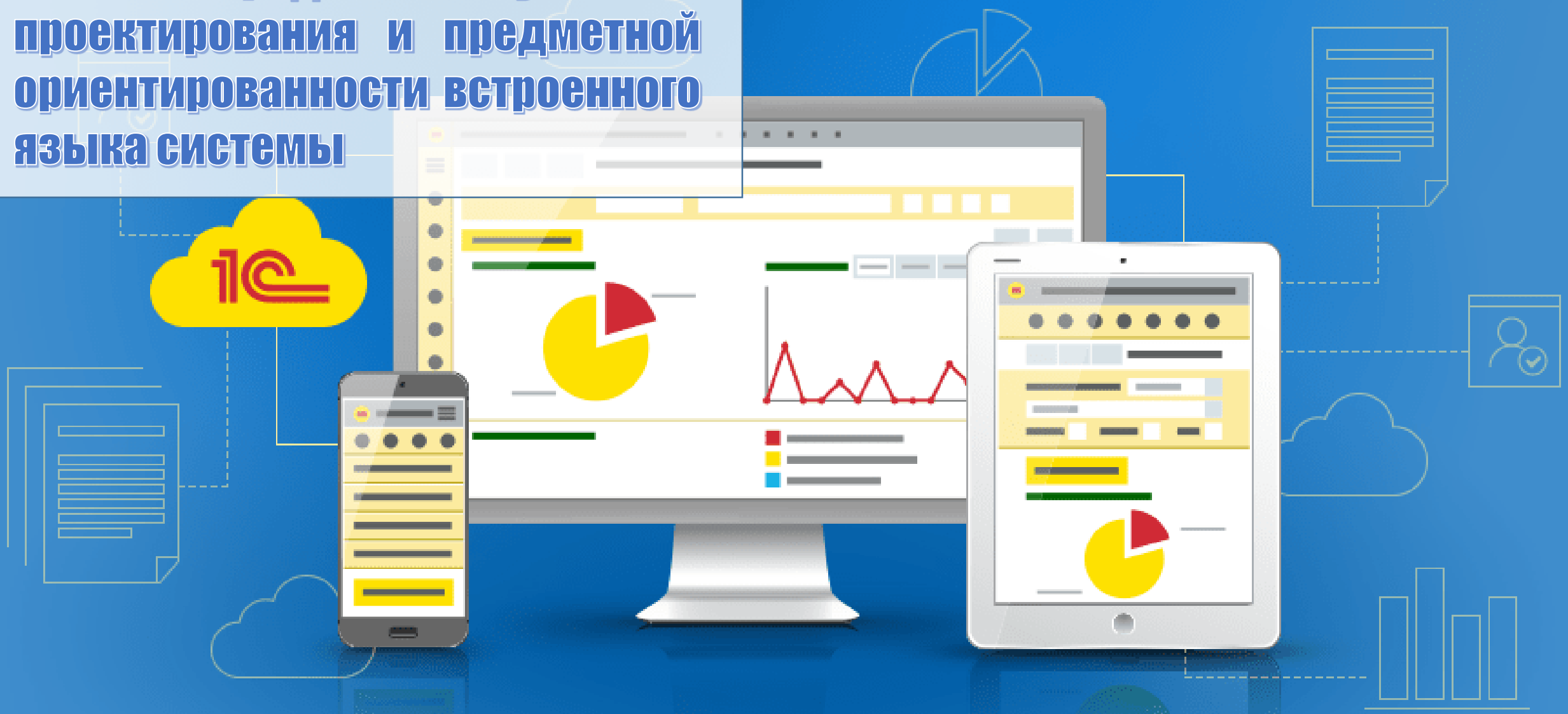
Рекомендации при проектировании структуры регистра

- При записи регистра сведений записываются все ресурсы. Не нужно проектировать структуру таким образом, что один ресурс меняется каждый день, а второй два раза в год. Лучше вынести второй ресурс в отдельный регистр сведений.
- Получение данных из нескольких ресурсов одного регистра сведений выполняется быстрее, чем получение одного ресурса из нескольких регистров.
- Запись в несколько ресурсов одного регистра сведений выполняется быстрее, чем запись одного ресурса в несколько регистров.

Возможные способы записи движений по регистру накопления

- ***При проведении документа***, т.е. в модуле объекта «Документ» посредством процедуры – обработчика событий «ОбработкаПроведения». Наиболее распространенный способ формирования движений в регистр.
- ***Из объекта документа, но без проведения.*** Любая процедура в модуле объекта документа может записать новые движения. Если это не процедура «ОбработкаПроведения» (и не вызывается из нее), тогда создается ситуация, при которой документ не проведен, но движения у него есть. Пример, процедура предварительного бронирования товара.
- ***Извне объекта документа.*** Любой объект, содержащий процедуру работы с набором записей регистра может приписать любому возможному регистратору данного регистра любые движения.
- ***Интерактивное внесение данных в регистр (ручная операция).*** Достаточно редко, но бывают ситуации, когда логику установки или изменения значений показателей в регистре по каким-то причинам сложно, не нужно или невозможно описать заранее, на этапе разработки конфигурации. Но при этом необходимо дать возможность пользователю все же данные в регистре изменять. В таких ситуациях обычно применяют интерактивный способ внесения информации в регистры. Реализуется посредством специального документа (регистратора), которому будет запрещено делать движения программным способом. Но при этом предоставляются интерактивные возможности (на форме) непосредственного создания записей в регистре. Никаких реквизитов и табличных частей документ одержать не должен.

Тема 2. Парадигма визуального проектирования и предметной ориентированности встроенного языка системы



- *Парадигма визуального проектирования и предметной ориентированности встроенного языка системы.*
- *Управляемое приложение.*
- *Декларативное описание пользовательского интерфейса.*
- *Клиентская и серверная части приложения.*
- *Директивы компиляции.*
- *Типы модулей и их назначение.*
- *Основные конструкции встроенного языка.*

Технология «Управляемое приложение»

- *Самостоятельное формирование структуры формы и размещение полей платформой. Если раньше разработчики описывали положение поля, указывая пиксели, то теперь есть возможность лишь указать вид группировки;*
- *Форма состоит из реквизитов, представляющих данные формы, и команд – выполняемых процедур и функций;*
- *Код формы выполняется на стороне и сервера, и клиента. Ведь сама по себе форма – это объект конфигурации, создаваемый на сервере и отображаемый на клиенте. Значит, объединяет в себе клиентскую и серверную часть;*
- *На клиентской стороне стали недоступны многие типы данных и теперь отсутствует возможность изменить данные в информационной базе;*
- *Для каждой процедуры или функции должна быть указана специальная настройка – директива компиляции.*

Директивы компиляции

&НаКлиенте (&AtClient)

Директива определяет выполнение процедуры (функции) на клиенте. Используется на клиенте, доступны процедуры модуля и доступны данные форм.

&НаСервере (&AtServer)

Директива определяет выполнение процедуры (функции) на сервере. Выполняется на серверном приложении. В таких процедурах доступны данные формы, доступен серверный контекст формы и вызовы серверных процедур модуля. Данные формы передаются с клиента на сервер и обратно по окончанию вызова.

&НаСервереБезКонтекста (&AtServerNoContext)

Директива определяет выполнение процедуры (функции) на сервере вне контекста формы. В данном случае не будут доступен контекст формы и ее данные. Позволяет вызывать только внекотекстные процедуры и функции и не позволяет выполнять передачу данных между клиентом и сервером. Данный метод позволяет существенно снизить объем передоваемой информации.

&НаКлиентеНаСервереБезКонтекста (&AtClientAtServerNoContext)

Директива определяет выполнение процедуры (функции) на сервере и на клиенте, не имеющую доступа к данным формы, переменным. В данном методе имеется доступ к процедурам и функциям клиентских и серверных одновременно.

&НаКлиентеНаСервере (&AtClientAtServer)

Директива определяет выполнение процедуры (функции) на сервере и на клиенте. В данном методе имеется доступ к процедурам и функциям общих модулей – серверных, не глобальных и серверных и клиентских одновременно, не имеющую доступ к переменным.

Особенности встроенного языка 1С

- предварительная компиляция – перед исполнением модули, содержащие текст на встроенном языке, преобразуются во внутренний код;
- кэширование скомпилированных модулей в памяти;
- мягкая типизация – тип переменной определяется типом значения, которое она содержит, и может изменяться в процессе работы;
- отсутствие программного описания объектов конфигурации – разработчик может использовать либо встроенные в платформу объекты, либо объекты, созданные системой в результате визуального конструирования прикладного решения.

Некоторые базовые конструкции встроенного языка

Конструкция	Описание
`Перем Адрес;`	Явное объявление переменной. Адрес - имя переменной. Начинаться имя переменной должно с буквы, либо с символа подчеркивания. Имя может состоять из букв, цифр и символов подчеркивания.
`А = 3;`	Переменную во встроенном языке 1С явно можно не объявлять. Первое присвоение значения этой переменной инициирует ее создание системой

`// Чтобы написать`
`// комментарий -`
`// ставятся две косые`
	Подсказки, пометки разработчика, которые помогают разобраться или вспомнить логику работы программного кода, прочая служебная информация может быть оформлена в виде комментариев. Каждая новая строка комментария начинется с символов //.

`НашеЧисло = 3.7+13*4;`	Переменной **НашеЧисло** присваивается числовое значение. С данными числового типа можно выполнять арифметические операции: сложение, вычитание, умножение и деление. В качестве разделителя целой и дробной части используется точка!
`С = -0.765;`	Числовые значения могут быть отрицательными.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание
ЭтоСтрока = "вуз";	Переменной ЭтоСтрока присваиваем строковое значение. Значение строкового типа пишется в кавычках
ФИО="Петров" + " " + "Семен" + " " + "АЛександрович"; //результат: ФИО = "Петров Семен Александрович"	Сложение строк – конкатенация.
ДатаОтчета = '2021.03.30';	Переменной присваивается значение типа Дата, которое всегда записывается в одинарных кавычках.
ЧислоСекунд = '2021.01.15' – '2021.01.17'; // ЧислоСекунд = 86400	Над датами можно производить операцию вычитания, в результате получим разницу между датами, измеренную в секундах. В сутках 86 400 секунд (60 сек * 60 мин * 24 ч).
НоваяДата ='2021.09.23'+ 86400; //НоваяДата = '2021.09.24'	К дате можно прибавлять и вычитать число. В результате к дате либо прибавится, либо отнимется число секунд.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание

Процедура ИмяПроцедуры
(ИмяПараметра1,
ИмяПараметра2,...)
// текст комментария
// тело процедуры
...
КонецПроцедуры
	За ключевым словом Процедура идет имя, затем указывают имена параметров, заключенных в круглые скобки. Между словами Процедура и КонецПроцедуры записывается тело процедуры. Имя должно начинаться с буквы или символа подчеркивания. Порядок описания (следования) процедур и функций между собой значения не имеет.

Функция ИмяФункции(Имя Параметра1, ...) // тело функции Возврат(Возвращаемое значение) ; КонецФункции	За ключевым словом Функция идет имя, затем указывают имена параметров, заключенных в круглые скобки. Далее идет тело функции. Функция должна возвращать результат в место ее вызова. Тело завершается ключевым словом КонецФункции.
Конструкции, реализующие ветвление, циклы.	
Если Оценка >= 3 Тогда Результат = "Экзамен сдан!"; Иначе Результат = "Еще надо подучить!"; КонецЕсли;	Простое условие. После слова КонецЕсли должна быть точка с запятой, потому что так заканчивается оператор Если.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание
Результат = ?(Оценка >= 3, "Экзамен сдан!", "Еще надо подучить!");	Сокращенное Если. Краткая запись предыдущего простого условия.
Если (Доход > 100000) И (КодКатегории = 2) Тогда КонецЕсли;	Составное логическое выражение.
Если Оценка >= 4 Тогда Результат = "Экзамен сдан, хорошие знания!"; ИначеЕсли Оценка = 3 Тогда Результат = "Сдан, но знания слабые"; Иначе Результат = "Знаний нет!"; КонецЕсли;	Множественное условие. Если первое условие не выполняется, то проверяется второе. Если ни одно из условий не выполняется то выполняется блок Иначе.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание
Пока Номер <= 20 Цикл КонецЦикла;	Простой цикл с неизвестным числом повторений. После слова КонецЦикла должна быть точка с запятой, потому что так заканчивается оператор Пока.
Для Номер = 1 По 20 Цикл КонецЦикла;	Простой цикл Для (цикл с известным числом повторений).
Для каждого СтрокаТаблицы Из Таблицы Цикл КонецЦикла;	Разновидность цикла для циклического обхода коллекций значений (универсальных коллекций значений (массивы, таблицы значений и т.п.) табличных частей справочников, документов и т.д.). При каждой итерации цикла возвращается новый элемент коллекции. Обход осуществляется до тех пор, пока не будут перебраны все элементы коллекции.

Некоторые базовые конструкции встроенного языка (продолжение)

Конструкция	Описание

Пока <условие> Цикл
  Если <условие> Тогда
  Продолжить;
  КонецЕсли;
КонецЦикла;
	Если необходимо передать управление в начало цикла, то используется оператор Продолжить.

Пока <условие> Цикл
  Если <условие> Тогда
  Прервать;
  КонецЕсли;
КонецЦикла;
	Если необходимо произвести досрочный выход из цикла, то используется оператор Прервать. В этом случае управление передается на операторы после цикла.

Наим=Стр.Наименование; Наим=Спр["Наименование"];	Два подхода, которые используются во встроенном языке при работе с объектными сущностями (объектами, с набором свойств, методов) для обращения к свойству объекта.
Спр.Печать();	Вызов методов объектов производится «через точку».

Пример структуры программного модуля

```
//*************** ОБЛАСТЬ ОБЪЯВЛЕНИЯ ПЕРЕМЕННЫХ *********************
Перем Фамилия Экспорт; //это глобальная переменная
Перем Имя, Отчество;     //это переменная модуля
Перем ФИО;       //это тоже переменная модуля и к ней можно обращаться
                 //из любой процедуры и функции нашего модуля

//************* ОБЛАСТЬ ОПИСАНИЯ ПРОЦЕДУР И ФУНКЦИЙ **************
Процедура Процедура1()
   Перем Итог;    //Итог это локальная переменная (переменная процедуры)
Итог = Фамилия+" "+Имя+" "+Отчество;

КонецПроцедуры
Функция Функция1()
   // операторы функции
Возврат(Фамилия + " "+ Имя);
КонецФункции
//****************** ОСНОВНОЙ ТЕКСТ ПРОГРАММЫ **********************
Фамилия ="Иванов";
Имя = "Иван";
Отчество = "Иванович";
//*****************************************************************************
```

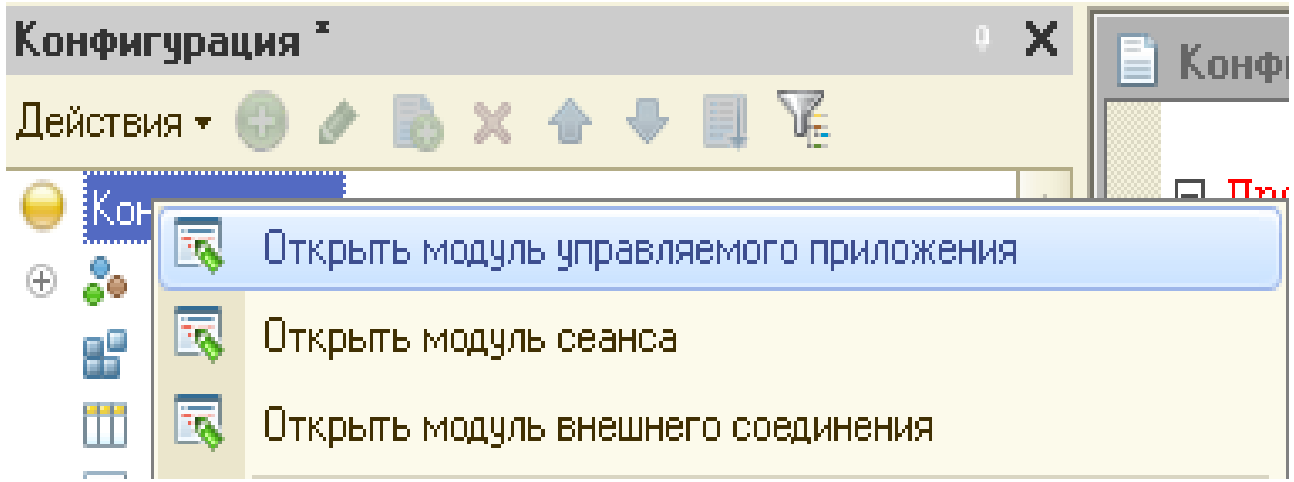
Типовая структура программного модуля включает:

- ***область объявления переменных.*** Размещается от начала текста модуля до первого оператора Процедура или оператора Функция или любого исполняемого оператора. В этом разделе могут находиться только операторы объявления переменных Перем;
- ***область описания процедур и функций.*** Размещается от первого оператора Процедура или оператора Функция до любого исполняемого оператора вне тела описания процедур или функций;
- ***основной текст программы.*** Размещается от первого исполняемого оператора вне тела процедур или функций до конца модуля. В этом разделе могут находиться только исполняемые операторы. Область основной текст программы исполняется в момент инициализации модуля. Обычно в разделе основной программы имеет смысл размещать операторы инициализации переменных какими-либо конкретными значениями, которые необходимо присвоить до первого вызова процедур или функций модуля

Виды модулей :

- Модуль приложения (управляемого или обычного);
- Модуль внешнего соединения
- Модуль сеанса
- Общие модули
- Модуль формы
- Модуль объекта
- Модуль менеджера объекта
- Модуль команды

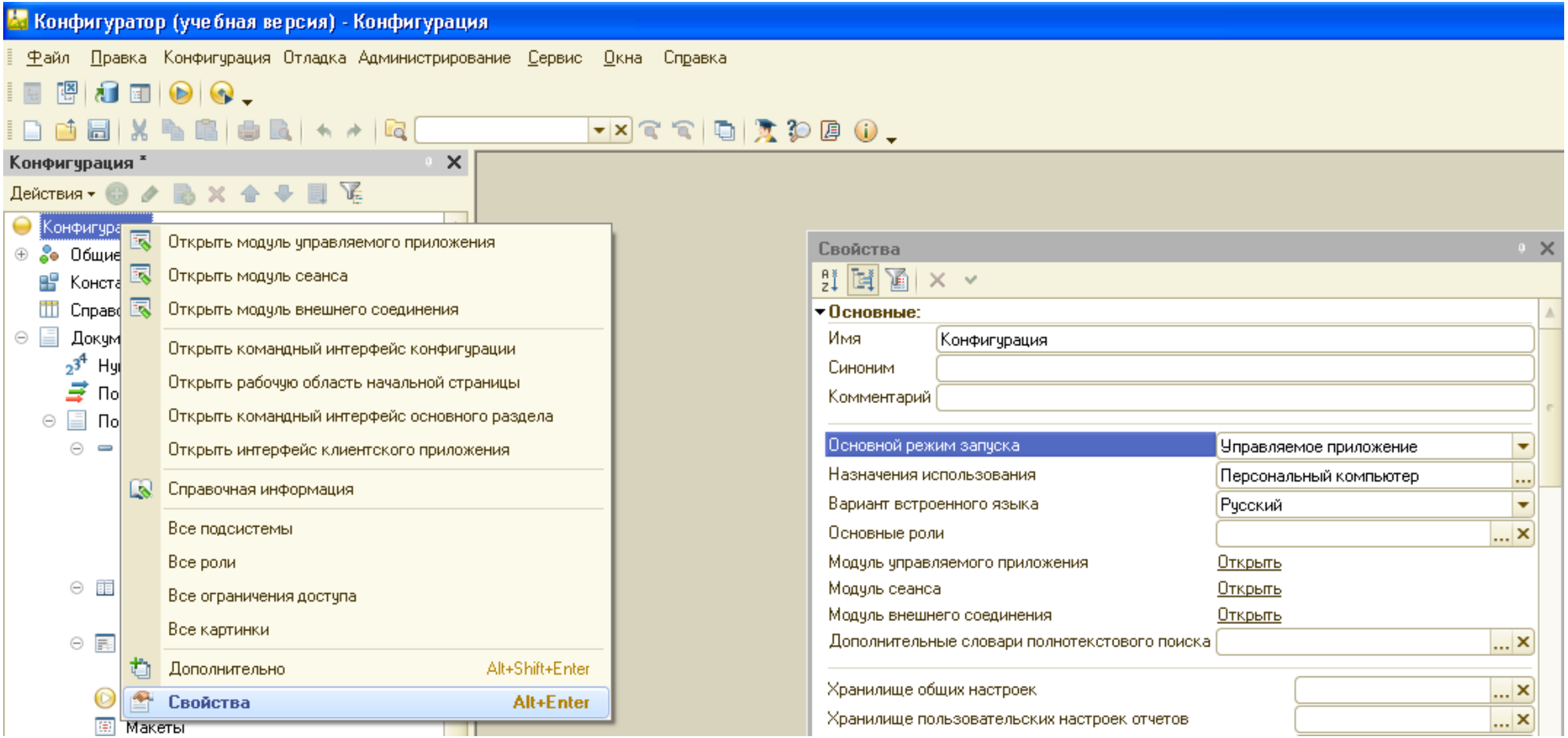
Модуль приложения (управляемого или обычного):



- может содержать все три области;
- выполняется на стороне клиента;
- располагается в корневом разделе конфигурации.

В модуле приложения описываются процедуры (обработчики) событий, которые инициализируются при старте и окончании работы системы. В платформе 1С 8 существует два различных модуля приложения: ***модуль Обычного приложения*** и ***модуль Управляемого приложения.*** Они срабатывают при запуске различных клиентов.

Настройка режима запуска приложения задается в свойстве конфигурации «Основной режим запуска»:



Модуль приложения обрабатывает события запуска и завершения приложения :

Конфигурация: Модуль управляемого приложения

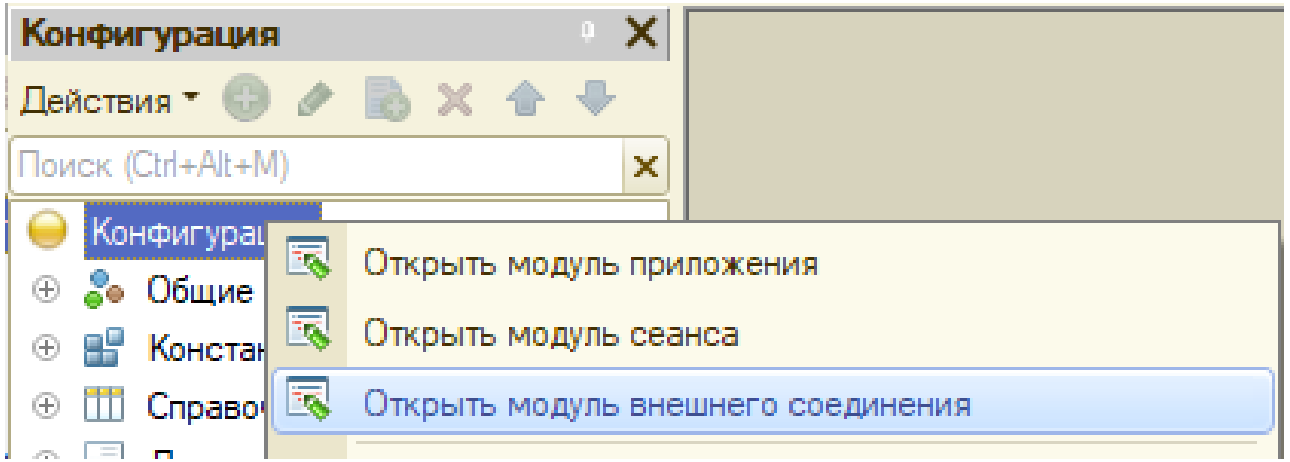
```
Процедура ПередНачаломРаботыСистемы(Отказ)
    // Вставить содержимое обработчика.
КонецПроцедуры

Процедура ПриНачалеРаботыСистемы()
    // Вставить содержимое обработчика.
КонецПроцедуры

Процедура ПередЗавершениемРаботыСистемы(Отказ)
    // Вставить содержимое обработчика.
КонецПроцедуры

Процедура ПриЗавершенииРаботыСистемы()
    // Вставить содержимое обработчика.
КонецПроцедуры
```

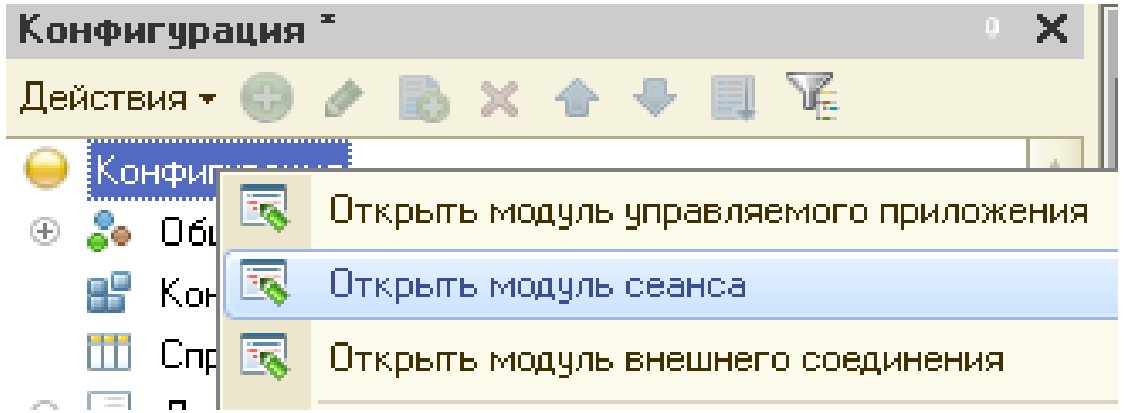
Модуль внешнего соединения:



- может содержать все три области;
- выполняется на стороне сервера;
- располагается в корневом разделе конфигурации.

В модуле внешнего соединения идет обработка событий старта и завершения работы приложения. Модуль внешнего соединения срабатывает, когда запуск приложения происходит в режиме com-соединения. Сам процесс внешнего соединения – это процесс не интерактивный. В этом режиме происходит программная работа с информационной базой и не происходит открытия окна приложения, что накладывает определенные ограничения на использование методов, предназначенных для интерактивной работы. В этом режиме нельзя использовать вызовы диалоговых форм, предупреждений и сообщений пользователю и т.п. Они просто не будут выполняться.

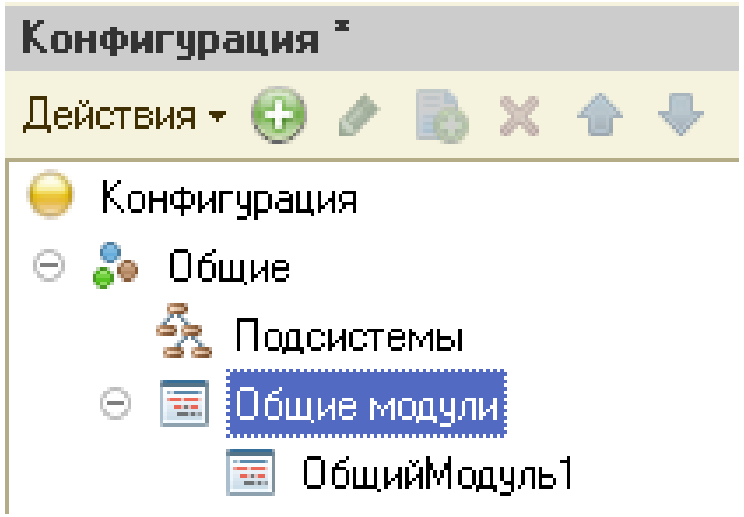
Модуль сеанса:



- может содержать область описания процедур и функций;
- выполняется на стороне сервера;
- располагается в корневом разделе конфигурации.

Это узкоспециализированный модуль, предназначенный исключительно для инициализации параметров сеанса. Его использование обусловлено тем, что само приложение может запускаться в различных режимах (что приводит к выполнению либо модуля управляемого, либо обычного приложения, либо модуля внешнего соединения), а инициализацию параметров сеанса необходимо производить вне зависимости от режима запуска. Чтобы не писать один и тот же программных код во всех трех указанных модулях, нам и потребовался дополнительный модуль, который выполняется вне зависимости от режима запуска приложения.

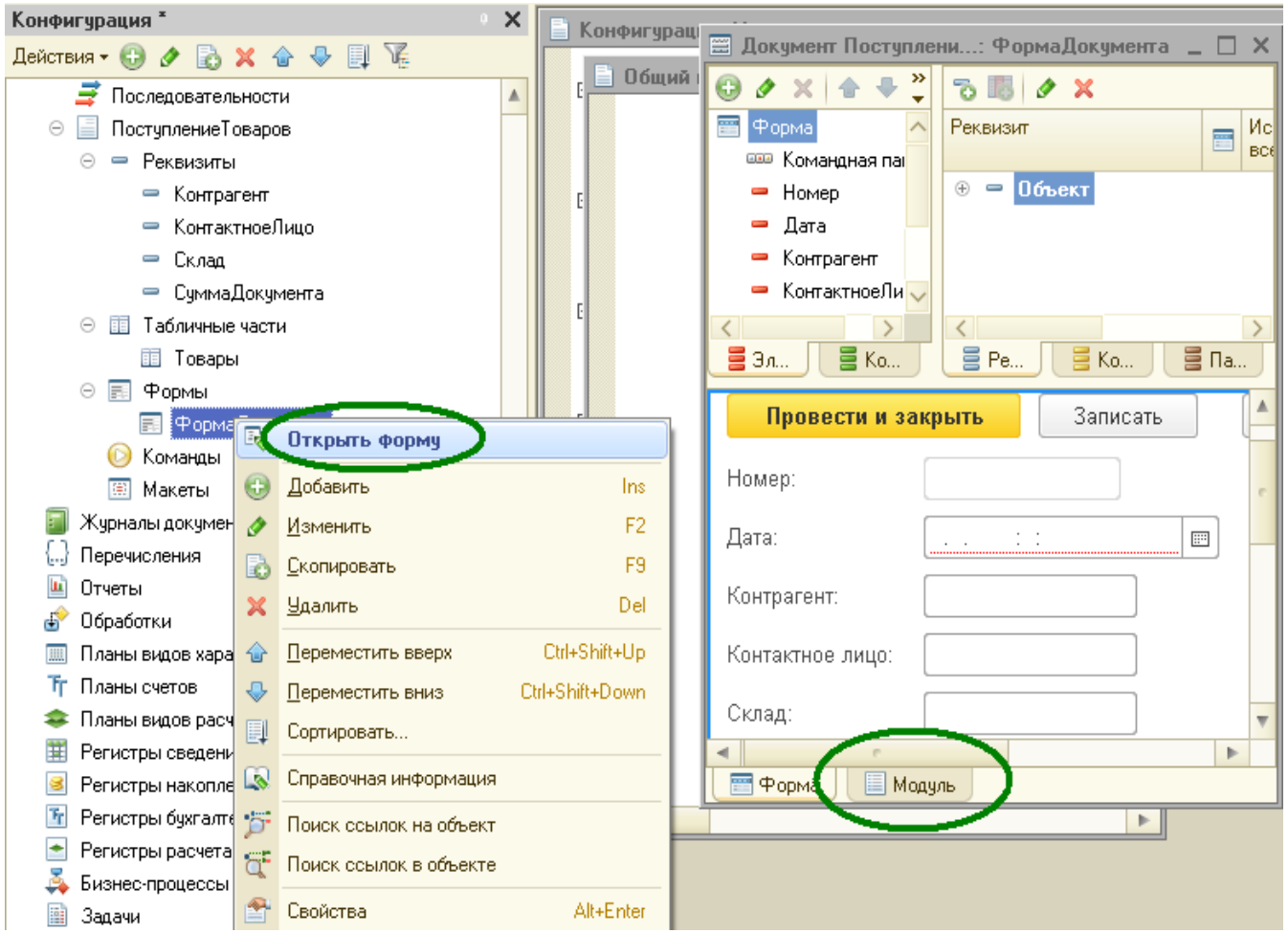
Общие модули:



- может содержать область описания процедур и функций;
- выполняется на стороне сервера или клиента (зависит от настроек модуля);
- располагается в ветке дерева объектов конфигурации «Общие» – «Общие модули»
- .

Общие модули предназначены для описания некоторых общих алгоритмов, которые будут вызываться из других модулей конфигурации. Общий модуль не содержит областей объявления переменных и основного текста программы. В нем можно объявлять экспортные методы, доступность которых будет определяться настройками модуля (на какой стороне он выполняется: на стороне сервера или клиента). В связи с тем, что раздел описания переменных не доступен, определять глобальные переменные в общих модулях нельзя. Для этого можно использовать модуль приложения.

Модуль формы:



- может содержать все 3 области;
- выполняется на стороне сервера и клиента.

Модуль формы предназначен для обработки действий пользователя с данной формой (обработка события нажатия кнопки, изменения реквизита формы и т.д.). Так же существуют события связанные непосредственно с самой формой (например, ее открытие или закрытие).

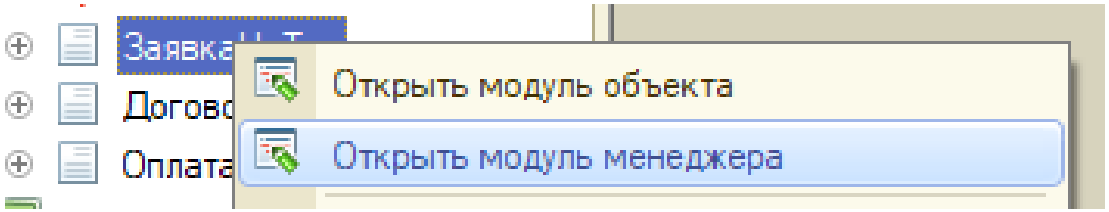
Модуль формы:

- может содержать все 3 области;
- выполняется на стороне сервера.

Данный модуль имеется у большинства объектов конфигурации и предназначен, в общем случае, для обработки событий, непосредственно связанных с объектом. Например, события записи и удаления объектов, проверка заполнения реквизитов объекта, проведение документа и т.д.

События модуля объекта будут вызываться в любом случае, даже в момент программной работы с объектом. Поэтому, если необходимо методы, связанные с объектом без привязки к конкретной форме объекта, то лучше использовать для этого модуль объекта.

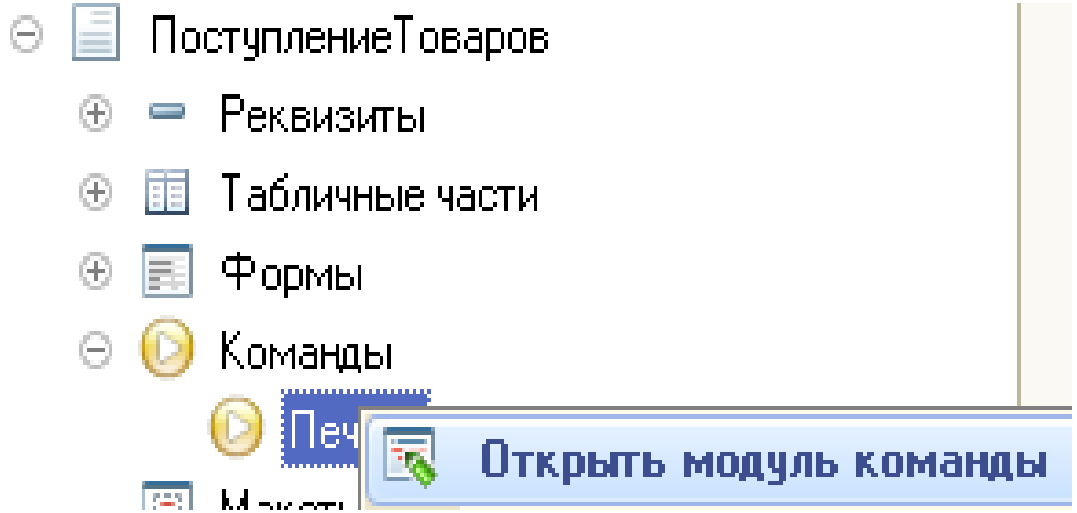
Модуль менеджера команды:



- может содержать все 3 области;
- выполняется на стороне сервера.

Модуль менеджера существует у всех прикладных объектов и предназначен для управления этим объектом как объектом конфигурации. Модуль менеджера позволяет расширить функциональность объекта за счет введения (написания) процедур и функций, которые относятся не к конкретному экземпляру объекта базы данных, а к самому объекту конфигурации. Модуль менеджера объектов позволяет размещать общие процедуры и функции для данного объекта и обращаться к ним из вне, например, из обработки (конечно, если эта процедура или функция будет с ключевым словом Экспорт).

Модуль команды:



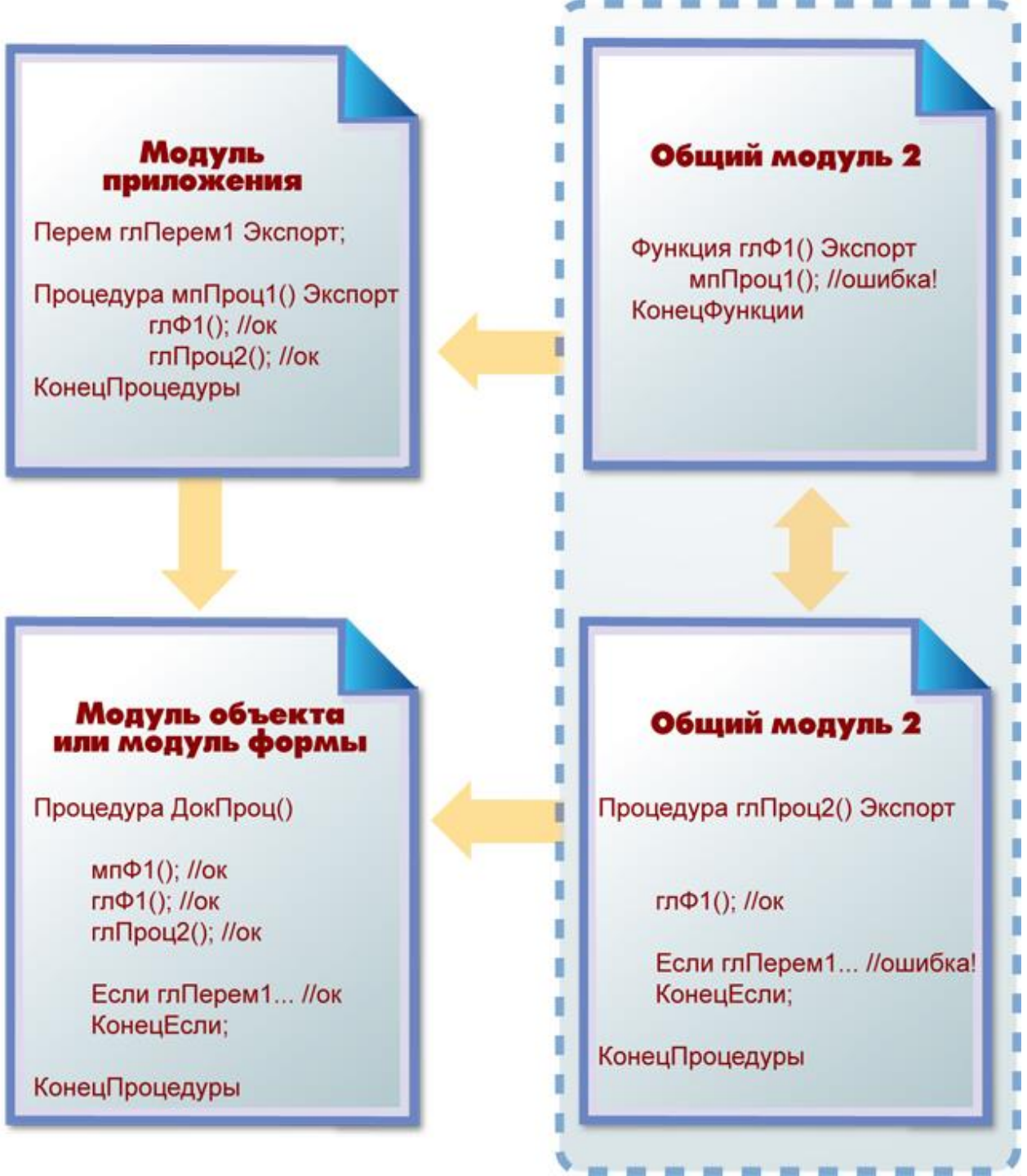
- может содержать раздел описания процедур и функций;
- выполняется на стороне клиента.

Команды – это объекты, подчиненные прикладным объектам или конфигурации в целом.

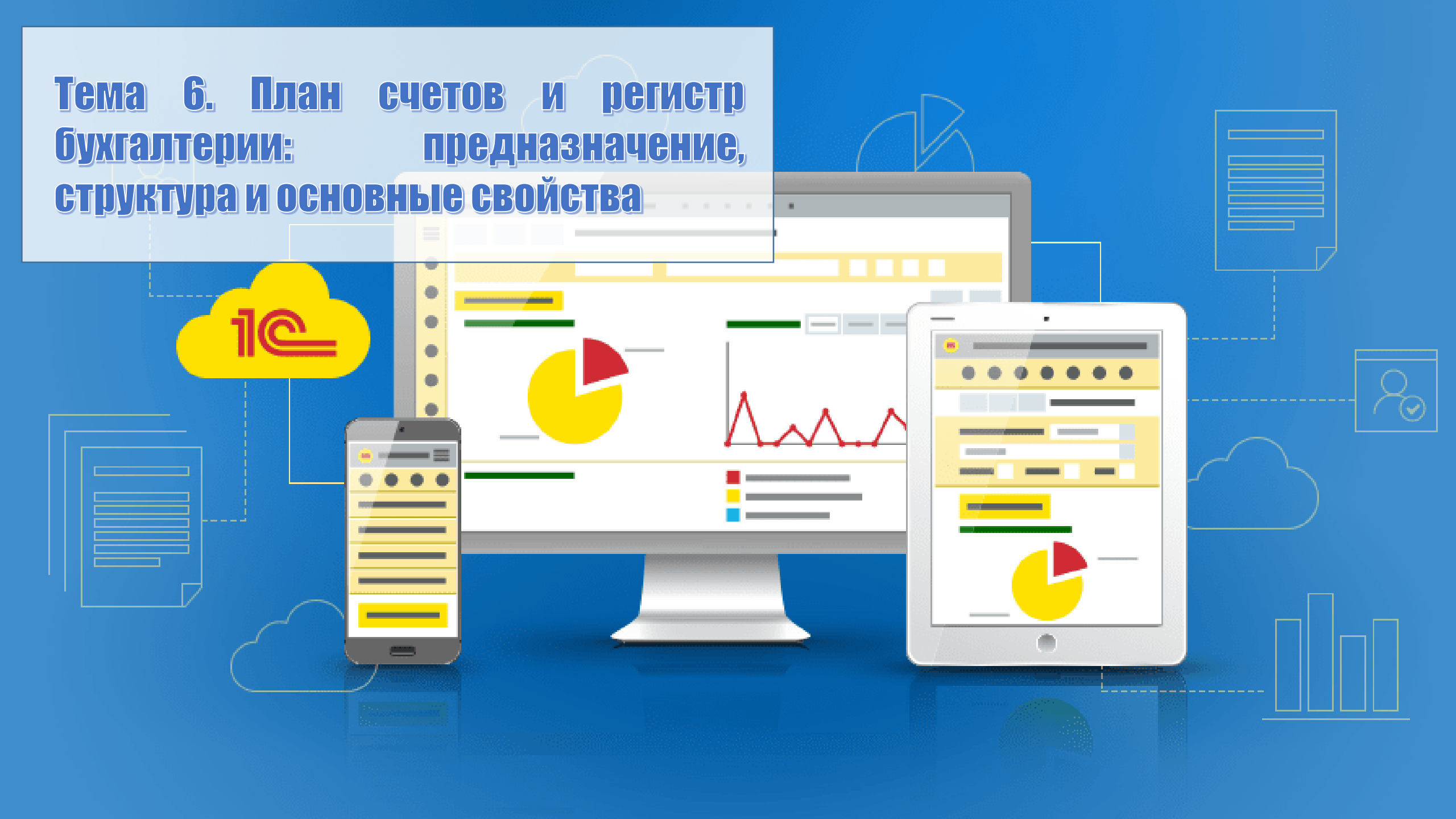
У каждой команды есть модуль команды, в котором можно описать предопределенную процедуру ОбработкаКоманды() для выполнения этой команды.

Правила видимости экспортируемых переменных, процедур и функций различных модулей:

1) В общем модуле недоступны экспортируемые переменные, процедуры и функции модуля приложения (модуля внешнего соединения).
2) В модуле приложения (модуле внешнего соединения) доступны экспортируемые процедуры и функции общих модулей.
3) В общих модулях доступны экспортируемые процедуры и функции других общих модулей.
4) В модулях прикладных объектов и модулях форм доступны экспортируемые переменные, процедуры и функции модуля приложения (модуля внешнего соединения), а также экспортируемые процедуры и функции общих модулей.
5) Если у формы назначен основной реквизит, то контекст модуля формы содержит дополнительные свойства и методы, связанные с основным реквизитом. Например, в модуле формы элемента справочника Номенклатура доступны свойства и методы объекта СправочникОбъект.Номенклатура.



Тема 6. План счетов и регистр бухгалтерии: предназначение, структура и основные свойства



- *План счетов и его основные свойства как объекта конфигурации.*
- *Предопределенные и пользовательские счета.*
- *Регистр бухгалтерии: предназначение, структура и основные свойства.*
- *Запись движений в регистр бухгалтерии.*
- *Итоги регистра бухгалтерии: физические таблицы регистра и расчет итогов. Разделение итогов регистра.*
- *Реальные и виртуальные таблицы регистра бухгалтерии для запросов.*

План счетов

- Является одним из основных понятий бухгалтерского учета.
- ***Планом счетов*** называется совокупность синтетических счетов, предназначенных для группировки информации о хозяйственной деятельности предприятия. Информация, накапливаемая на таких синтетических счетах, позволяет получить полную картину состояния средств предприятия в денежном выражении.
- Все счета, входящие в один баланс, объединяются в один план счетов. По свой сути план счетов – это справочник счетов.

План счетов

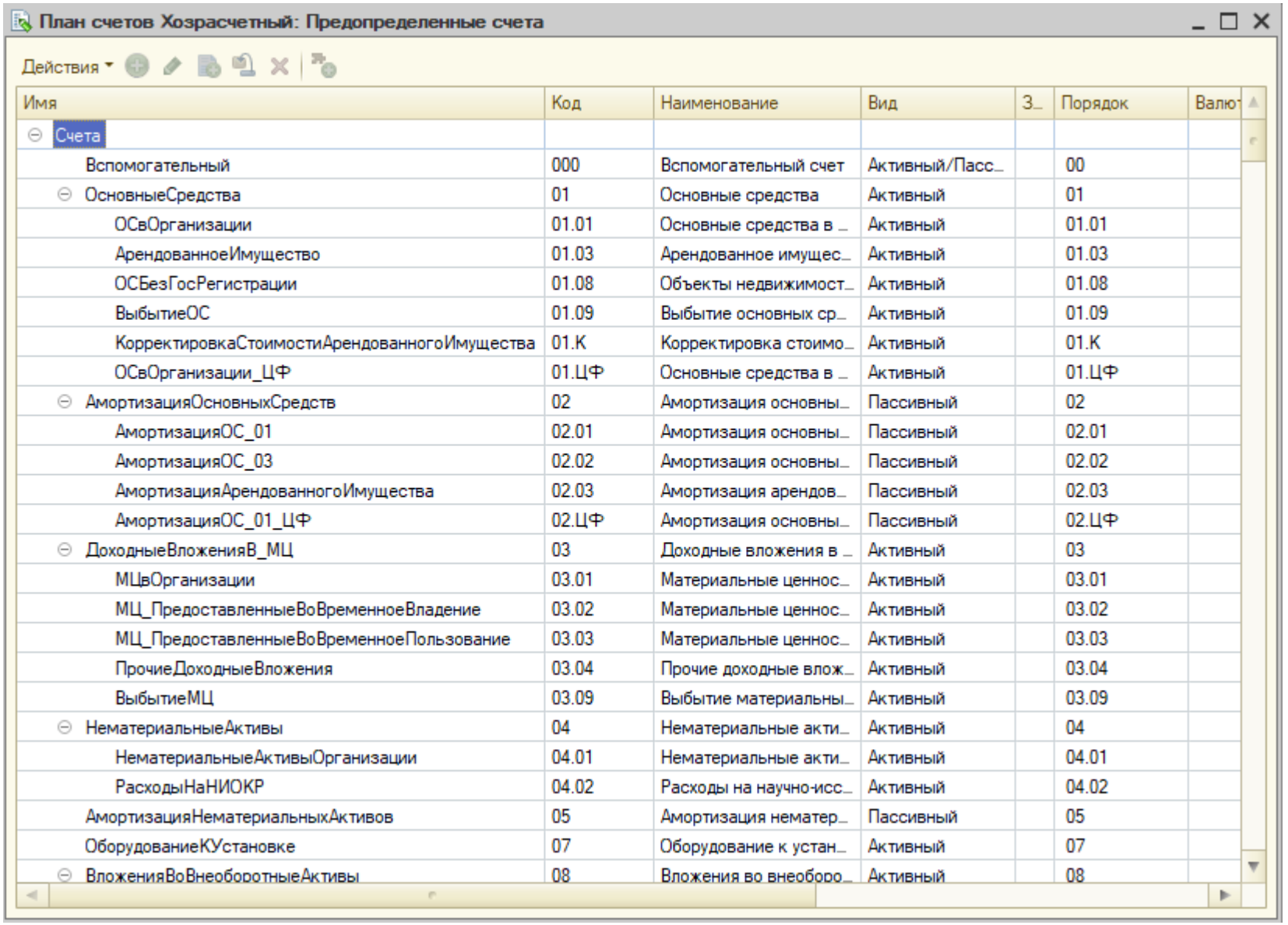
- Система «1С:Предприятие» предоставляет гибкие возможности по ведению планов счетов. Собственно путем настройки плана счетов и организуется требуемая система учета.
- В системе «1С:Предприятие» может быть ***несколько планов счетов*** и учет по всем планам счетов можно вести одновременно. Общее число планов счетов, которое может быть организовано в системе, с технической точки зрения неограниченно и определяется исключительно реальными потребностями учета.

Субсчета

- Планы счетов в системе «1С:Предприятие» поддерживают ***многоуровневую иерархию*** «счет - субсчета».
- Каждый план счетов может включать ***неограниченное*** число счетов первого уровня.
- К каждому счету может быть открыто также неограниченное количество ***субсчетов***. В свою очередь, каждый субсчет может иметь свои субсчета и так далее.
- Количество уровней субсчетов в системе «1С:Предприятие» ***неограниченно.***

Предопределенные счета и счета, введенные пользователем

Для плана счетов ввод счетов в режиме Конфигуратор является в большинстве случаев основным вариантом заполнения. План счетов, ну или по крайней мере его основа (счета первого уровня), создается еще на этапе конфигурирования и во многом определяет логику учета. Ввод новых счетов (субсчетов) пользователем является скорее исключением, чем правилом.

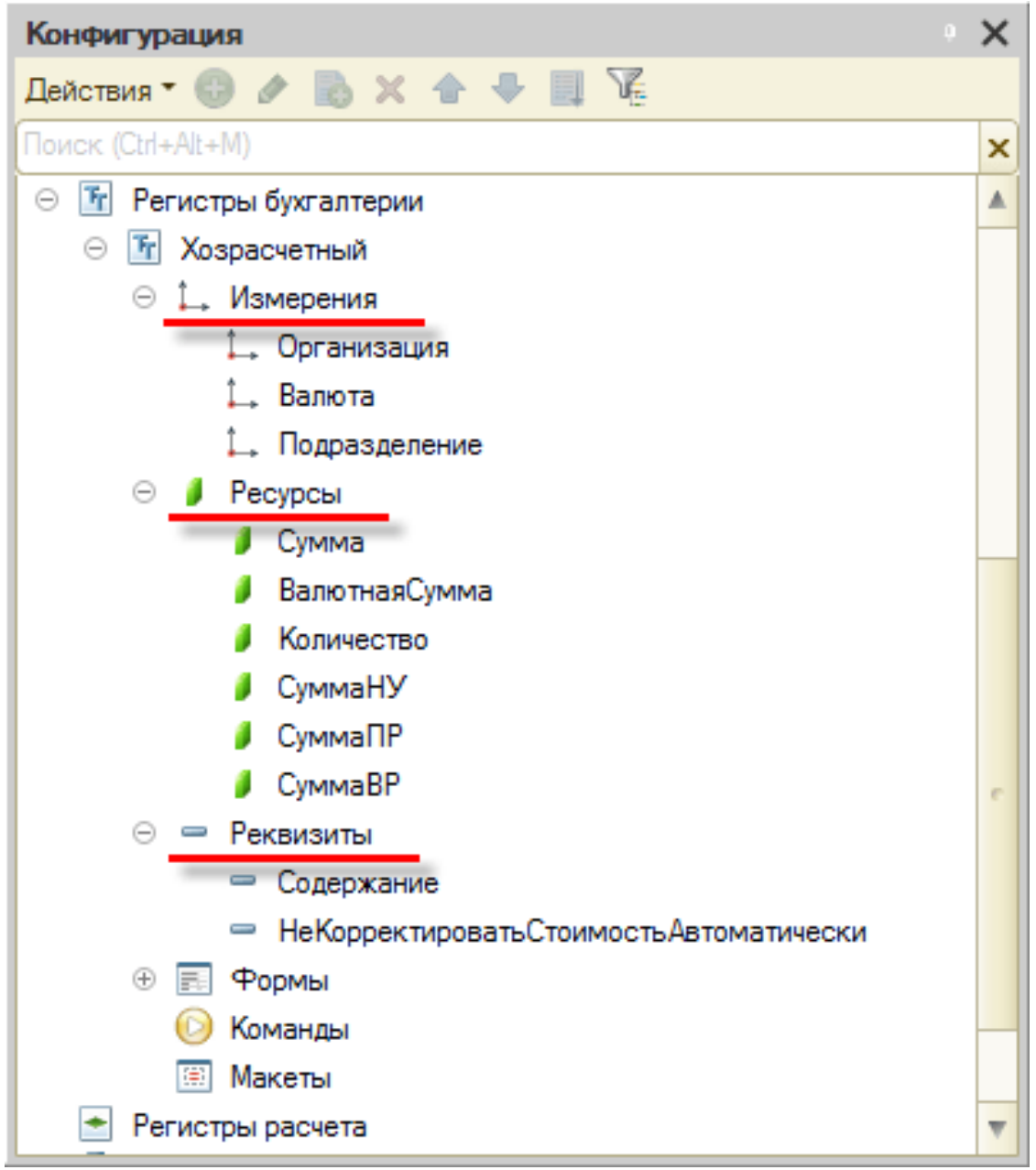

План счетов Хозрасчетный: Предопределенные счета

Действия

Имя	Код	Наименование	Вид	З...	Порядок	Валют
Счета						
Вспомогательный	000	Вспомогательный счет	Активный/Пасс...		00	
ОсновныеСредства	01	Основные средства	Активный		01	
ОСвОрганизации	01.01	Основные средства в ...	Активный		01.01	
АрендованноеИмущество	01.03	Арендованное имущес...	Активный		01.03	
ОСБезГосРегистрации	01.08	Объекты недвижимост...	Активный		01.08	
ВыбытиеОС	01.09	Выбытие основных ср...	Активный		01.09	
КорректировкаСтоимостиАрендованногоИмущества	01.К	Корректировка стоимо...	Активный		01.К	
ОСвОрганизации_ЦФ	01.ЦФ	Основные средства в ...	Активный		01.ЦФ	
АмортизацияОсновныхСредств	02	Амортизация основны...	Пассивный		02	
АмортизацияОС_01	02.01	Амортизация основны...	Пассивный		02.01	
АмортизацияОС_03	02.02	Амортизация основны...	Пассивный		02.02	
АмортизацияАрендованногоИмущества	02.03	Амортизация арендов...	Пассивный		02.03	
АмортизацияОС_01_ЦФ	02.ЦФ	Амортизация основны...	Пассивный		02.ЦФ	
ДоходныеВложенияВ_МЦ	03	Доходные вложения в ...	Активный		03	
МЦвОрганизации	03.01	Материальные ценнос...	Активный		03.01	
МЦ_ПредоставленныеВоВременноеВладение	03.02	Материальные ценнос...	Активный		03.02	
МЦ_ПредоставленныеВоВременноеПользование	03.03	Материальные ценнос...	Активный		03.03	
ПрочиеДоходныеВложения	03.04	Прочие доходные влож...	Активный		03.04	
ВыбытиеМЦ	03.09	Выбытие материальны...	Активный		03.09	
НематериальныеАктивы	04	Нематериальные акти...	Активный		04	
НематериальныеАктивыОрганизации	04.01	Нематериальные акти...	Активный		04.01	
РасходыНаНИОКР	04.02	Расходы на научно-исс...	Активный		04.02	
АмортизацияНематериальныхАктивов	05	Амортизация нематер...	Пассивный		05	
ОборудованиеКУстановке	07	Оборудование к устан...	Активный		07	
ВложенияВоВнеоборотныеАктивы	08	Вложения во внеоборо...	Активный		08	

Регистры бухгалтерии

- Для отражения в бухгалтерском учете информации о хозяйственных операциях в системе «1С:Предприятие» используются **регистры бухгалтерии**, описываемые в ветви дерева конфигурации Регистры бухгалтерии.
- При разработке структуры регистра: создаются наборы ***измерений, ресурсов*** и ***реквизитов регистра***;



Регистр бухгалтерии – это «регистр регистров»

- По своей сути регистр бухгалтерии – это «регистр регистров», совокупность регистров.
- Один регистр бухгалтерии содержит в себе множество учетных регистров (счетов), каждый из которых может иметь независимую аналитику (субконто), в разрезе которой и будут отражаться учетные показатели.
- В целях контроля все учетные регистры (счета) связаны между собой правилом двойной записи, которое образует замкнутую систему показателей: невозможно изменить значение показателя одного учетного регистра, не затронув другой, причем на ту же сумму.

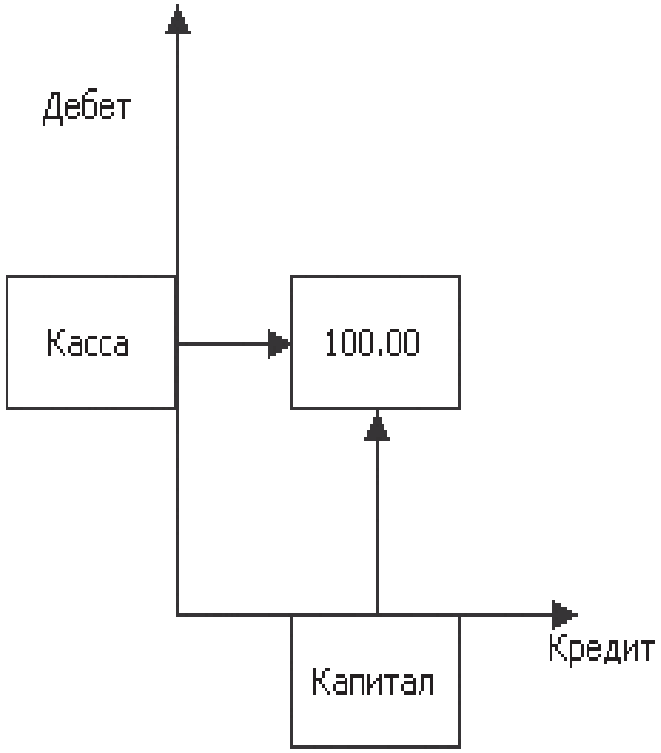
Ресурсы регистра бухгалтерии

- Если единственным обязательным разрезом ведения бухгалтерского учета являются счета и субсчета, то единственное значение, которое в этом разрезе нужно учитывать, – это ***сумма в валюте учета*** (или валюте составления баланса, можно сказать и так).
- Для ***регламентированного учета*** этой валютой является ***национальная валюта***. Другими словами, для ведения регламентированного бухгалтерского учета достаточно добавить в регистр бухгалтерии один ресурс, который обычно называют ***Сумма***, в который будут записываться значения, выраженные в рублях.

Реализация принципа двойной записи в регистре бухгалтерии

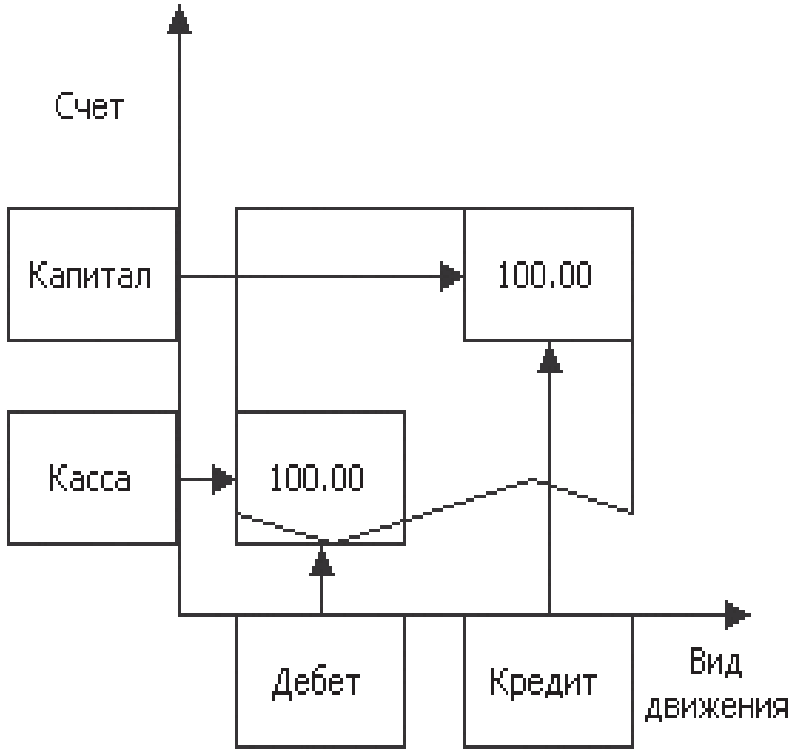
Регистр бухгалтерии с поддержкой корреспонденции

№	Дебет	Кредит	Сумма
1	Касса	Капитал	100.00



Регистр бухгалтерии без поддержки корреспонденции

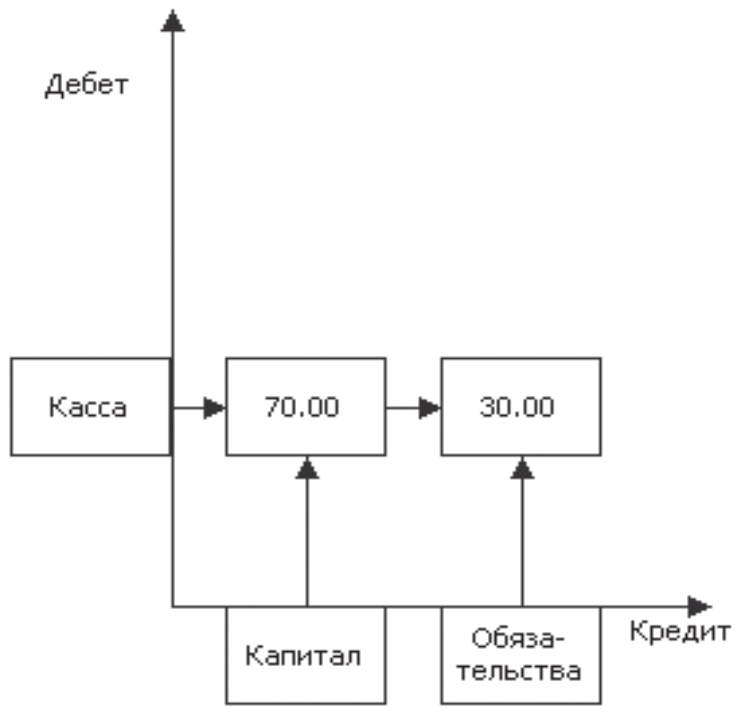
№	Счет	Вид движения	Сумма
1	Касса	Дебет	100.00
2	Капитал	Кредит	100.00



Контроль двойной записи

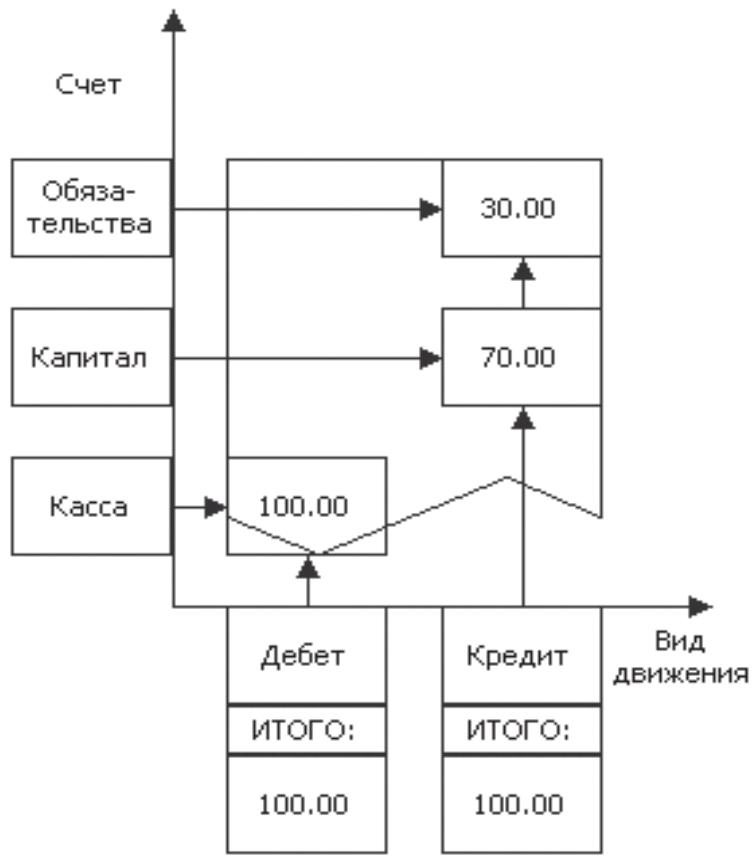
Регистр бухгалтерии с поддержкой корреспонденции

№	Дебет	Кредит	Сумма
1	Касса	Капитал	70.00
2	Касса	Обяза-тельства	30.00



Регистр бухгалтерии без поддержки корреспонденции

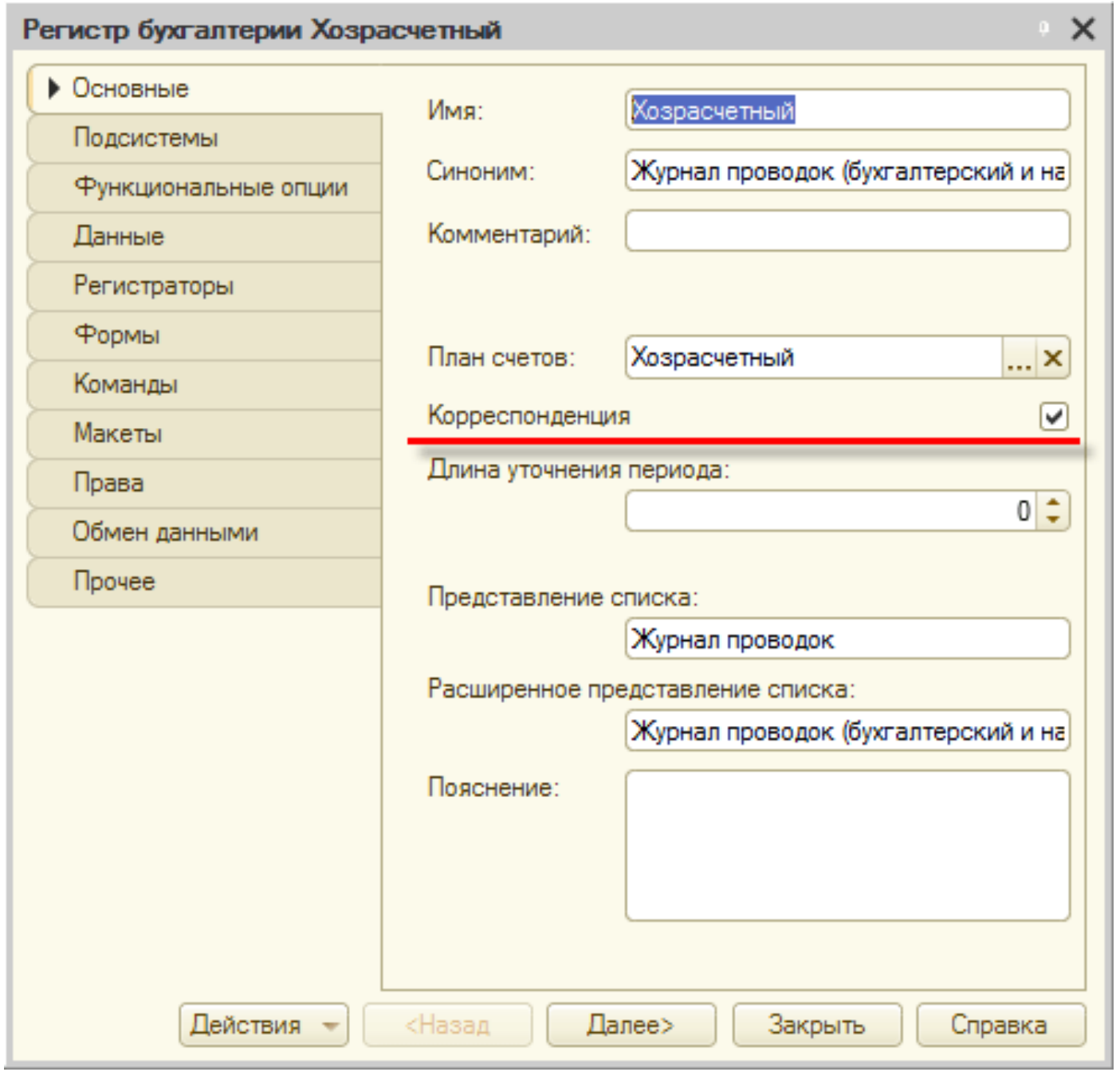
№	Счет	Вид движения	Сумма
1	Касса	Дебет	100.00
2	Капитал	Кредит	70.00
3	Обяза-тельства	Кредит	30.00



- Если речь идет об автоматизации ***нерегламентированного учета***, то в качестве валюты управленческого учета стараются выбрать наиболее устойчивую валюту или валюту той страны, в которой расположены основные деловые партнеры (учредители, инвесторы). В РФ управленческий учет чаще всего ведется в долларах США и с недавних пор в евро.
- Ресурс регистра бухгалтерии может иметь только ***числовой тип***, его длиной и точностью мы можем управлять.

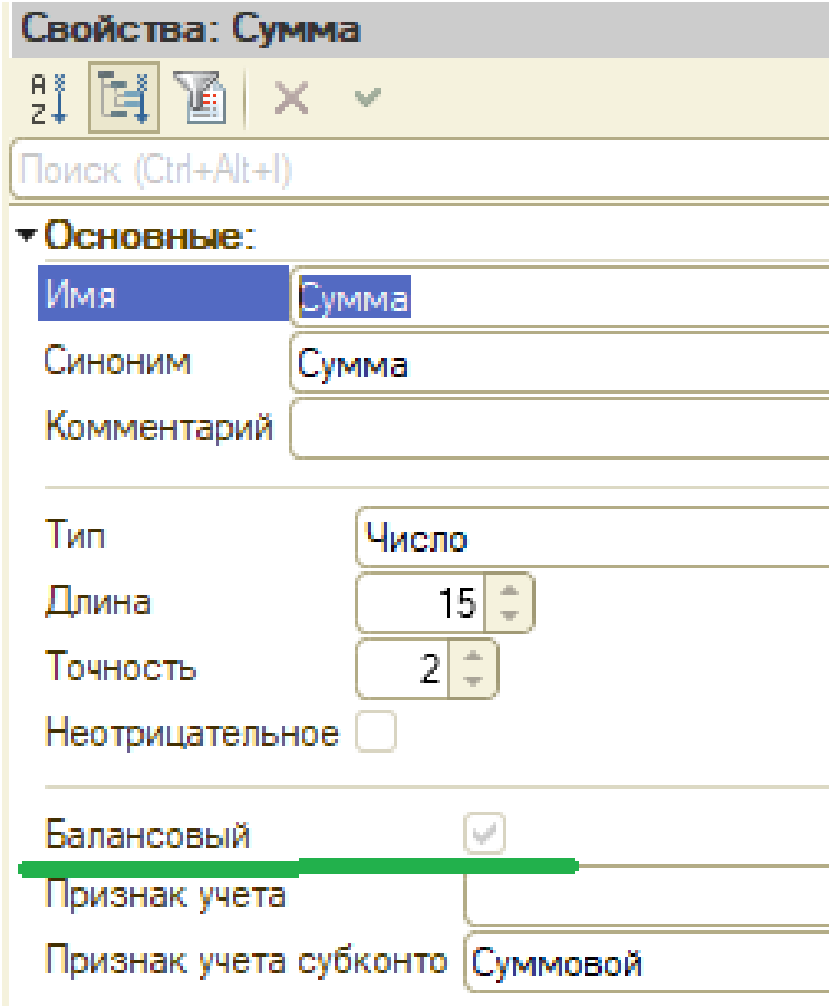
Поддержка корреспонденции

На закладке Основные выбирается план счетов и ***признак поддержки корреспонденций.***



Свойство ресурса «Балансовый»

- ***Свойство Балансовый*** влияет на структуру регистра бухгалтерии. Основная цель бухгалтерского учета – баланс.
- Основывается баланс на методе двойной записи, в соответствии с которым при движении в учете каждое учитываемое значение (в данном случае сумма в валюте учета) должно «проходить» по двум сторонам проводки: по дебету и по кредиту.
- При установке свойства Балансовый платформа создаст в регистре бухгалтерии с поддержкой корреспонденции одно новое свойство – Сумма. Таким образом, заполняя сумму проводки, пользователь указывает и сумму, на которую изменяется дебетуемый счет проводки, и ту (ее же), на которую изменяется кредитуемый.



Свойство измерения «Балансовый»

- Балансовые измерения делят весь учет (все разделы учета) на самостоятельные балансы. Баланс, как известно, – замкнутая система показателей. Например, балансовое измерение Организация – по каждой организации программой будет составлен свой баланс. Это значит, что у каждой организации будут свои активы (касса, расчетные счета, склады и др.), свои обязательства (перед сотрудниками, перед поставщиками) и свой собственный капитал.
- Наличие балансового измерения в регистре бухгалтерии совершенно исключает возможность, например, переместить товары с одной организации в другую. По каждой организации ведется свой баланс. А значит, для решения задачи перемещения товаров нужно в одной организации их списать (на какой-нибудь счет расчетов с той, другой, организацией), а в другой организации их оприходовать (со счета учета расчетов с первой организацией).
- Балансовое измерение добавляет новое поле во все таблицы регистра бухгалтерии.
- При записи набора движений в регистр осуществляется контроль двойной записи по каждому значению (сочетанию значений, если балансовых измерений несколько) балансового измерения.

Небалансовое измерение регистра бухгалтерии

- по своей сути напоминает субконто. Так же как и субконто, оно делит не все разделы учета на самостоятельные балансы, а, как правило, лишь некоторые, или все разделы учета, но баланса по нему получить нельзя.
- Для регистра с поддержкой корреспонденции добавляются два поля – по одному на каждую сторону проводки, например, ВалютаДт и ВалютаКт.
- Для регистра без поддержки поле добавляется одно, но при записи набора движений в регистр контроль двойной записи по каждому значению такого измерения не осуществляется.
- Небалансовое измерение можно сравнить с субконто, которое в обязательном порядке прикрепили ко всем счетам.
- Хотя хранение в физических таблицах регистра у них различное, функционально механизмы небалансовых измерений и субконто похожи

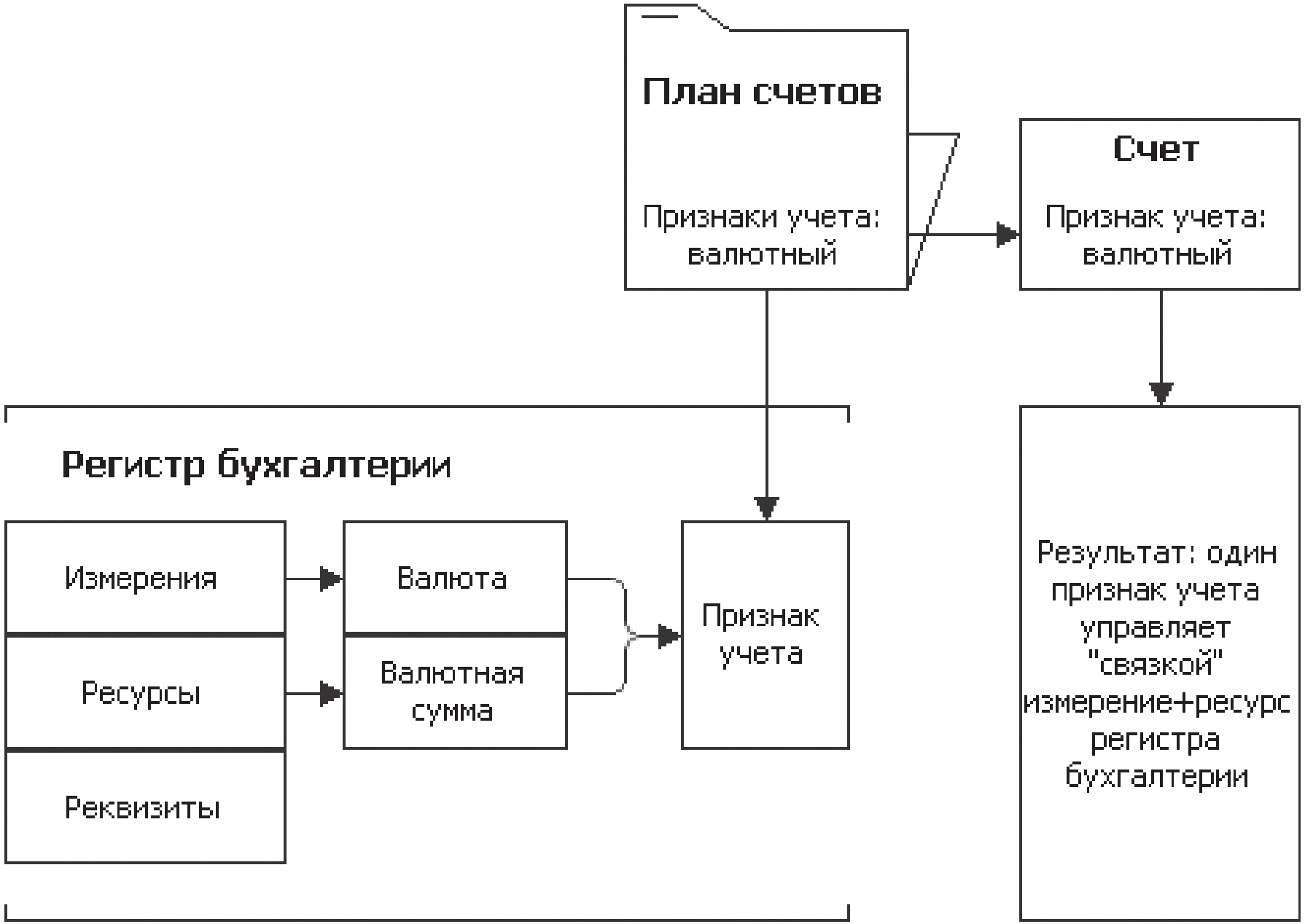
Сравнение небалансовых измерений и субконто

Критерий	Небалансовое измерение	Субконто
Компактность хранения в базе	Поле резервируется даже для счетов, где измерение не используется	Резервируется только в том случае, если субконто на счете используется
Тип данных	Любой, чаще просто ссылка на справочник	Всегда составной, причем состав определятся для плана видов характеристик
Возможность включить/отключить для конкретного счета	Да, с помощью признаков учета	Да, при подключении вида субконто на счет
Всегда на своем месте	Да	Может быть любым по порядку на счете, и для пользовательских счетов порядок может быть задан произвольно пользователем

Сравнение субсчетов, субконто и небалансовых измерений

Решение	Преимущества	Недостатки
1	2	3
Субсчета	Может потребовать незначительного изменения типового решения, если оно с самого начала предусматривало возможность расширения субсчетов второго или третьего уровня	Сложность получения отчетности по одному источнику финансирования и сложность ведения учета из-за значительного числа счетов в плане счетов
Субконто	Может потребовать незначительного изменения типового решения, если уложиться в максимальное число субконто для плана счетов	Может потребовать значительных изменений, если все субконто для отдельных счетов уже заняты и нужно увеличивать максимальное число субконто. Увеличение максимального числа субконто может негативно повлиять на производительность
Небалансовое измерение	Позволит разделить учет более оптимально, чем в случае субконто, и удобно подготавливать отчетность, как по значению измерения, так и сводно	Потребует значительных изменений типового решения

Схема взаимодействия объектов



Взаимодействие объектов в рамках валютного учета

- Для реализации схемы потребуется еще один связанный с ним объект – периодический регистр сведений Курсы валют.
- Регистр будет иметь измерение Валюта типа Ссылка на справочник Валюты и ресурс Курс типа Число, а если требуется вести учет в валютах с очень маленькими значениями курса, то два ресурса – Курс и Кратность, и для получения эквивалента в валюте учета валютную сумму нужно будет сначала умножить на курс, а затем разделить на кратность.
- Необходимо добавить новый признак учета в плане счетов Валютный. Он позволит отметить те счета, которые требуют ведения валютного учета.
- Нужно добавить новый ресурс регистра бухгалтерии ВалютнаяСумма, где будет храниться сумма в валюте.
- Добавить новое небалансовое измерение регистра бухгалтерии Валюта – ссылка на элемент справочника Валюты, по которому будет «разделен» учет отмеченного признаком счета.
- Результат: при установке одного флажка в настройках счета для счета становятся доступными для заполнения новый ресурс и новое измерение.

Функциональные возможности регистра бухгалтерии

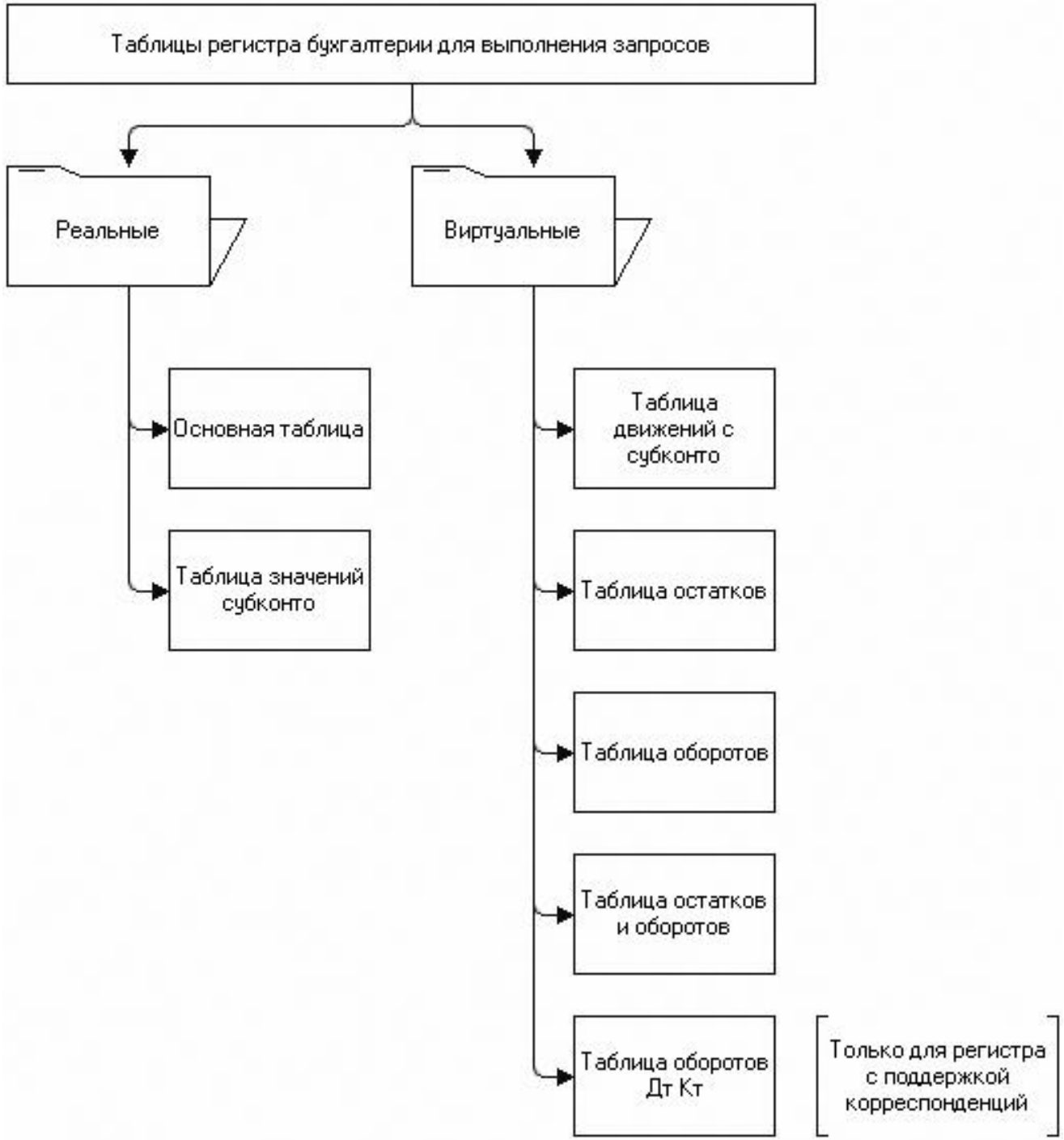
Итак, основными функциональными возможностями, которые предоставляет регистр бухгалтерии разработчику, являются:

- выбор записей в заданном интервале по заданным критериям;
- выбор записей по регистратору;
- получение остатков и оборотов на указанный момент времени по заданным значениям параметров (счет, субконто, измерения, кор. счет, кор. субконто, кор. измерения);
- получение остатков на указанный момент времени по заданным значениям параметров (счетДт, субконтоДт, счетКт, субконтоКт, измерениеКт, измерения (для балансовых) и измерениеДт, измерениеКт (для не балансовых));

Функциональные возможности регистра бухгалтерии (продолжение)

- режим работы с разделением итогов, который обеспечивает более высокую параллельность записи в регистр;
- отключение использования текущих итогов;
- расчет итогов на указанную дату;
- чтение, изменение и запись набора записей в регистр;
- возможность записи в регистр без пересчета итогов;
- полный пересчет итогов и пересчет итогов за указанный период.

Реальные и виртуальные таблицы регистра бухгалтерии для запросов



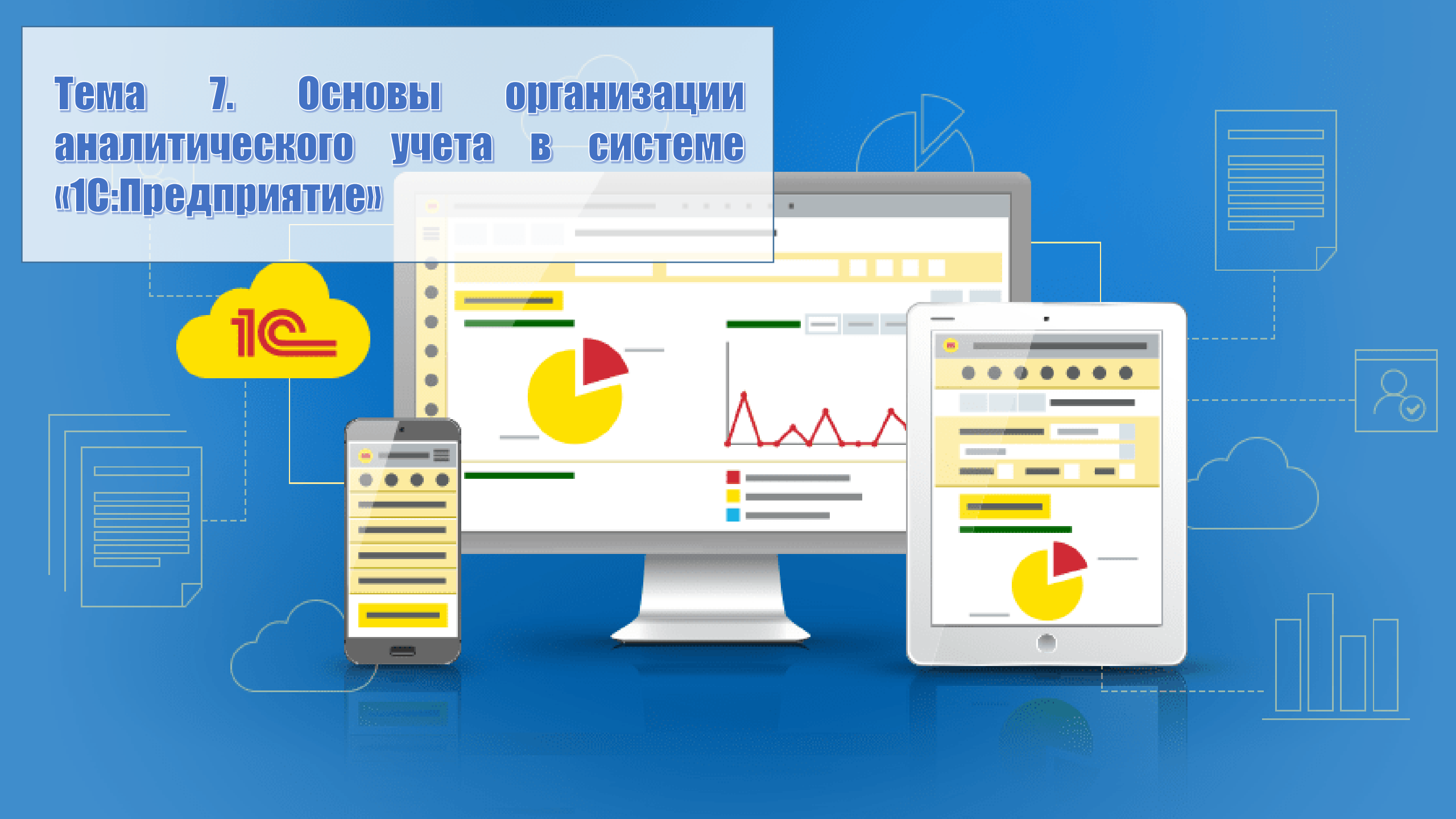
Параметры виртуальных таблиц регистра бухгалтерии

Параметр	Таблица оборотов	Таблица оборотов Дт Кт	Таблица остатков	Таблица остатков и оборотов	Таблица движений с субконто
Период, НачалоПериода, КонецПериода	Да	Да	Да	Да	Да
Условие	Да	Да	Да	Да	Да
УсловиеСчета, УсловиеКорСчета, УсловиеСчетаДт, УсловиеСчетаКт	Да	Да	Да	Да	
Субконто, КорСубконто, СубконтоДт, СубконтоКт	Да	Да	Да	Да	
Периодичность	Да	Да		Да	
МетодДополнения				Да	

Поля виртуальных таблиц регистра бухгалтерии

Параметр	Таблица оборотов	Таблица оборотов Дт Кт	Таблица остатков	Таблица остатков и оборотов	Таблица движений с субконто
Счет	Да		Да	Да	
КорСчет	Да				
Счет Дт/Кт		Да			Да
<Измерение>	Да	Да	Да	Да	Да
<Измерение>Кор	Да				
<Измерение>Дт/Кт		Да			Да
Субконто<N>	Да		Да	Да	
КорСубконто<N>	Да				
СубконтоДт<N>/Кт<N>		Да			Да
Период	Да*	Да*		Да*	Да
Регистратор	Да*	Да*		Да*	Да
НомерСтроки	Да*	Да*		Да*	Да
МоментВремени					Да
Активность					Да
<Реквизит>					Да
ВидСубконтоДт<N>/Кт<N>					Да
<Ресурс>					Да
<Ресурс> Дт/Кт					Да
<Ресурс>Остаток			Да	Да**	
<Ресурс>ОстатокДт/Кт			Да	Да**	
<Ресурс>РазвернутыйОстатокДт/Кт			Да	Да**	
<Ресурс>Оборот	Да	Да		Да	
<Ресурс>ОборотДт/Кт	Да	Да		Да	
<Ресурс>КорОборот	Да				
<Ресурс>КорОборотДт/Кт	Да				

Тема 7. Основы организации аналитического учета в системе «1С:Предприятие»



- *Основы организации аналитического учета в системе "1С:Предприятие": сквозная аналитика, обычная аналитика, опционная аналитика.*
- *Организация обычной аналитики с помощью субконто.*
- *План видов характеристик и виды субконто.*
- *Настройка плана счетов для аналитического учета.*

Иерархическая аналитика

- Иерархический план счетов позволяет организовать детальный учет. И при желании можно организовать детальный аналитический учет, используя возможность организации иерархического плана счетов и ввода новых субсчетов пользователем.
- Однако эта детализация будет иерархической. При такой организации аналитического учета не представляется возможным получение итогов по нескольким параллельным срезам.
- Рассмотрим учет товаров на складах. Предположим, в нашей организации есть несколько складов, на каждом из которых множество видов номенклатурных позиций. Перед нами стоит задача обеспечить получение остатков и оборотов по каждому складу и товару в отдельности и сводно по всем. Задачу можно решить, используя субсчета счета Товары. Изменив длину кода счета (до 10 символов) и маску кода счета @@.@@.@.@@, мы можем завести к счету учета товаров необходимое количество субсчетов.
- Подобная организация аналитического учета возможна, но имеет ряд существенных недостатков. Во-первых, план счетов будет расти (и расти существенно) по мере изменения номенклатуры товаров (добавления новых позиций, отказа от старых и появления неиспользуемых). Во-вторых, он будет расти нелинейно, т. к. одна и та же позиция номенклатуры может храниться на разных складах; таким образом, количество субсчетов на каждую позицию будет равняться количеству складов. И самое главное – не представляется возможным автоматически сгруппировать данные по одной позиции номенклатуры, хранимой на разных складах.

Аналитический учет

- Под ***аналитическим учетом*** будем понимать учет более детальный, чем синтетический, т.к. аналитические отчеты, как правило, являются более подробными, чем синтетические, или, наоборот, синтетические отчеты можно считать сводными относительно отчетов аналитических.

3 варианта реализации аналитического учета

- ***сквозная аналитика***, при которой один и тот же аналитический разрез используется на всех или почти на всех счетах;
- ***обычная аналитика***, при которой на разных счетах должна присутствовать различная аналитика в разном количестве параллельных срезов;
- ***опциональная аналитика***, которая является разновидностью обычной, но может добавляться или убираться со счетов пользователя без участия программиста. Используется механизм функциональных опций.

Сквозная аналитика

- Организуется с помощью небалансовых измерений регистра бухгалтерии и признаков счета для управления доступностью измерения дебета/кредита записи регистра в зависимости от настройки выбранного счета дебета/кредита проводки.

Опциональная аналитика

- если конфигурация разрабатывается с расчетом на множество организаций, которые должны ее использовать с минимальными изменениями, тогда чем больше вариантов настройки программист вынес на опции, которыми можно управлять пользователем, тем она лучше. Такими опциями, например, может управляться настройка аналитического учета на некоторых счетах.

Обычная аналитика

- Это детализация учета таким образом, что на разных счетах есть возможность установить разные аналитические разрезы и разное их количество (на одном может вообще не быть аналитики, другой может иметь один, два или более параллельных срезов).
- Для создания такого механизма необходимо:
  - хранить где-то список возможных аналитических разрезов, причем каждый разрез (вид субконто, аналитический счет) должен помнить тип своих значений;
  - описать максимально возможное количество срезов для каждого регистра бухгалтерии (плана счетов);
  - указать, на каких счетах, какие разрезы будут использоваться и в какой последовательности.

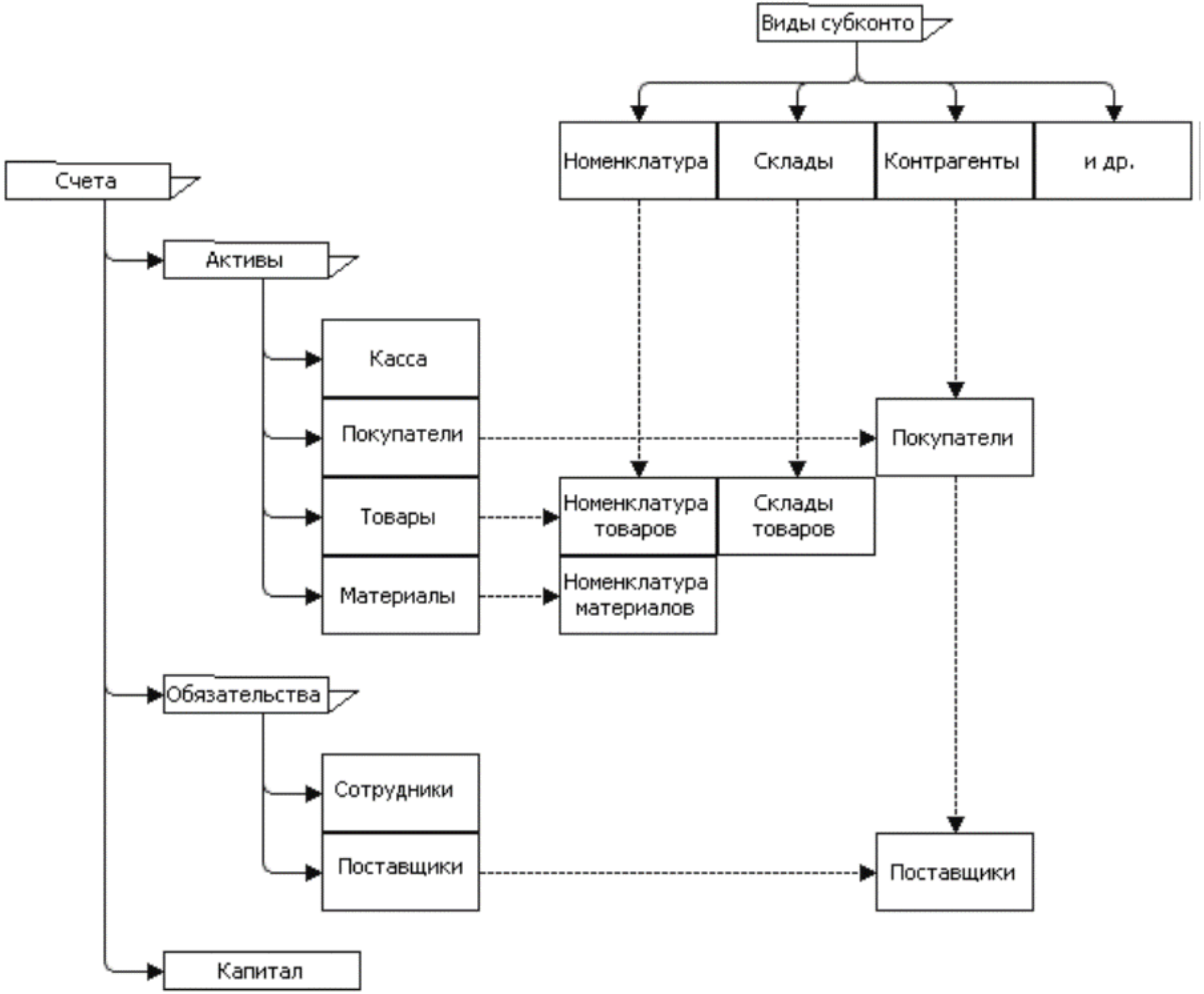
Понятия Вид субконто и Субконто

- ***Вид субконто*** – группа однородных объектов аналитического учета.
- Понятие «вид субконто» соответствует понятию бухгалтерского учета «***аналитический счет***».
- Каждый элемент списка вида субконто представляет собой объект аналитического учета и называется ***Субконто***.

Понятия Вид субконто и Субконто (продолжение)

- Ни субконто, ни вид субконто не имеют объектного представления в системе и существуют лишь как понятия, реализуемые с помощью других объектов.
- Виды субконто, как правило, бывают ссылочного типа и содержат в себе ссылку на справочник, документ или перечисление, хотя возможны и другие варианты.
- Если видом субконто является справочник, то субконто – один элемент этого справочника.

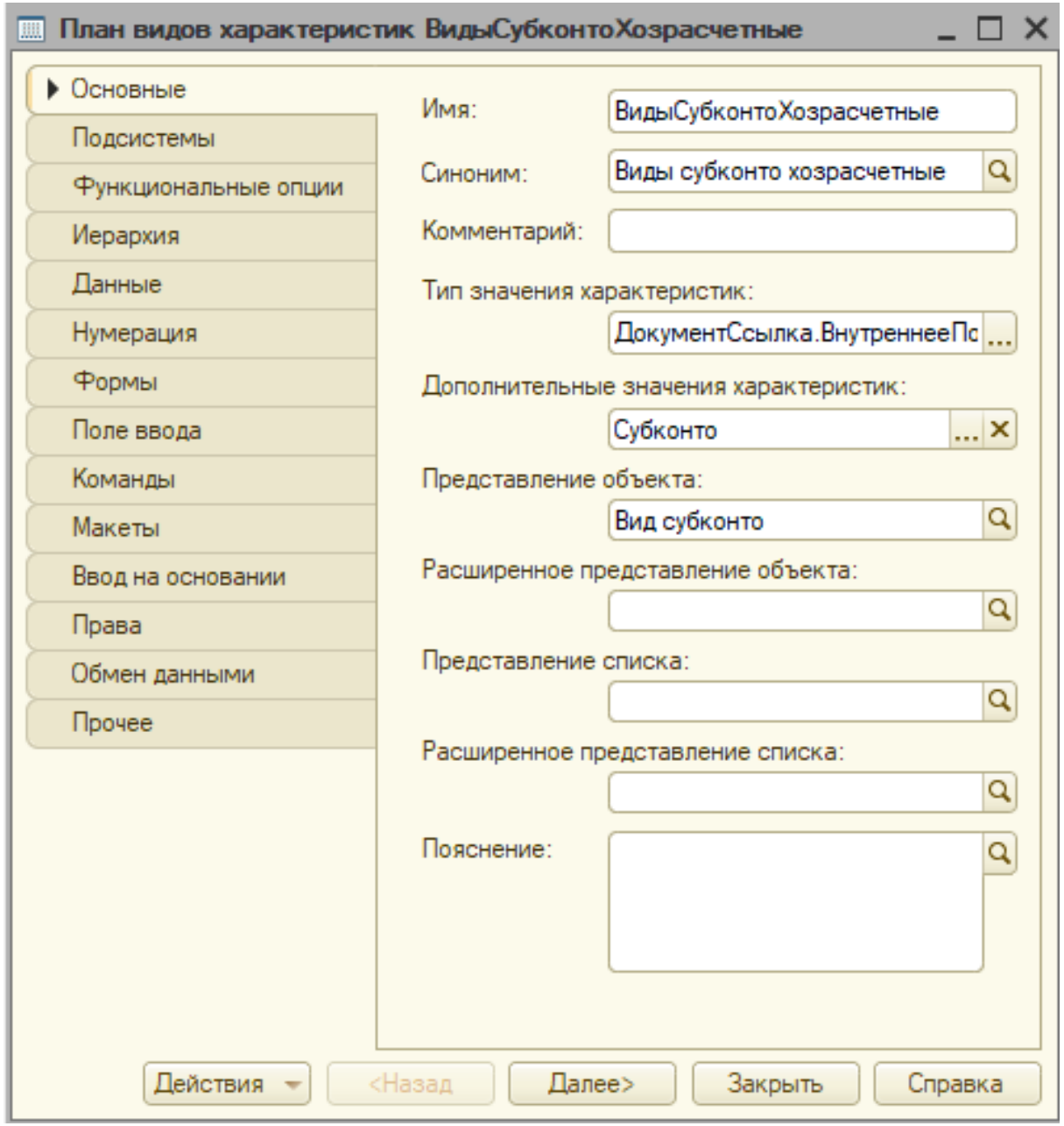
Использование видов субконто для организации аналитического учета



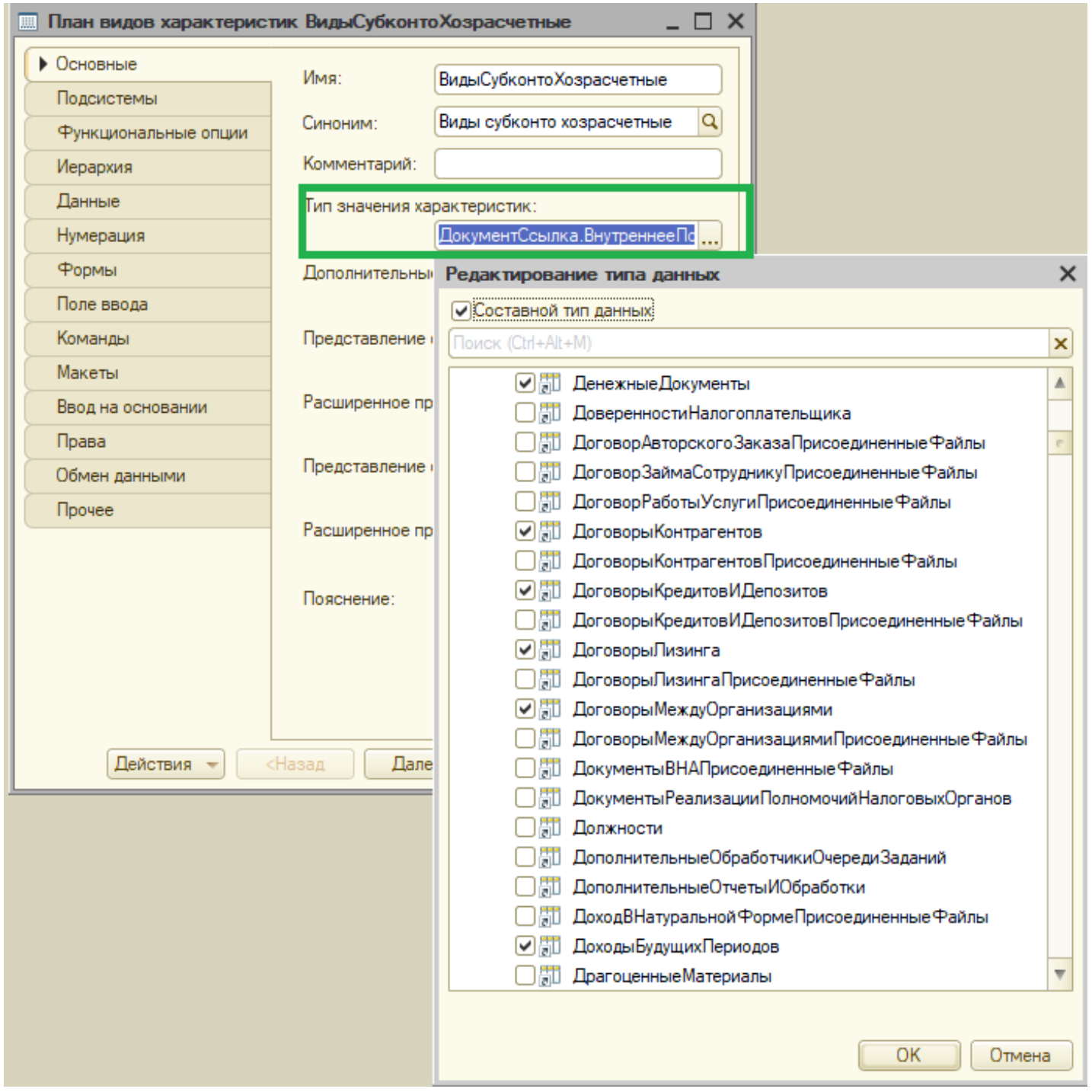
Соответствие понятий системы «1С:Предприятие» и понятий предметной области

Объект	Понятие
Планы видов характеристик	Все виды субконто
Вид характеристик	Вид субконто
Характеристика	Субконто

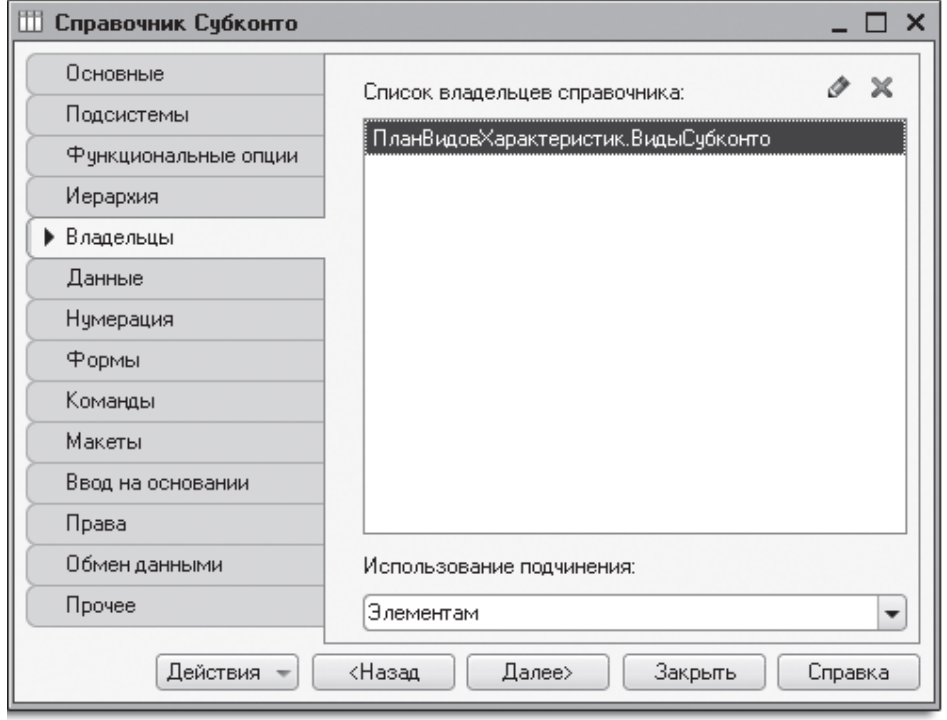
Создание плана видов характеристик для субконто

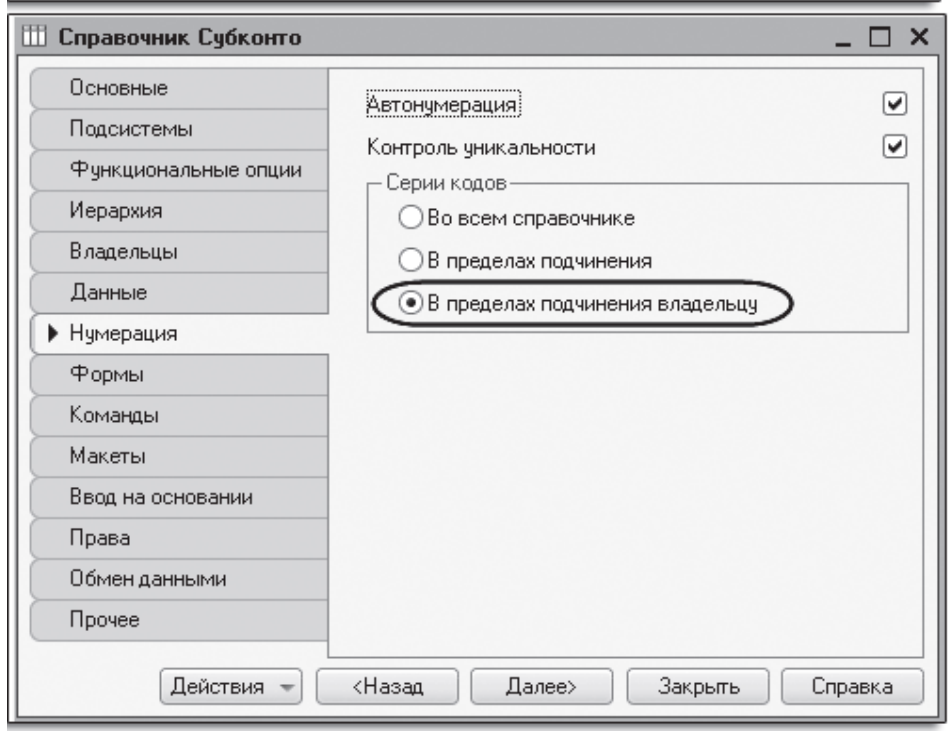


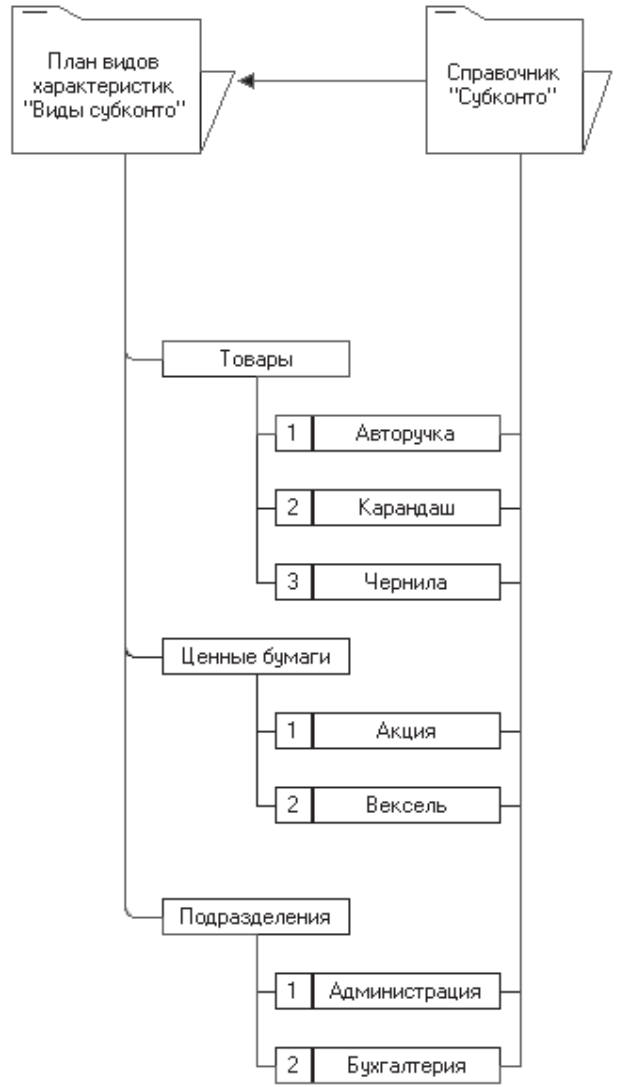
Настройка составного типа значения характеристик – видов субконто



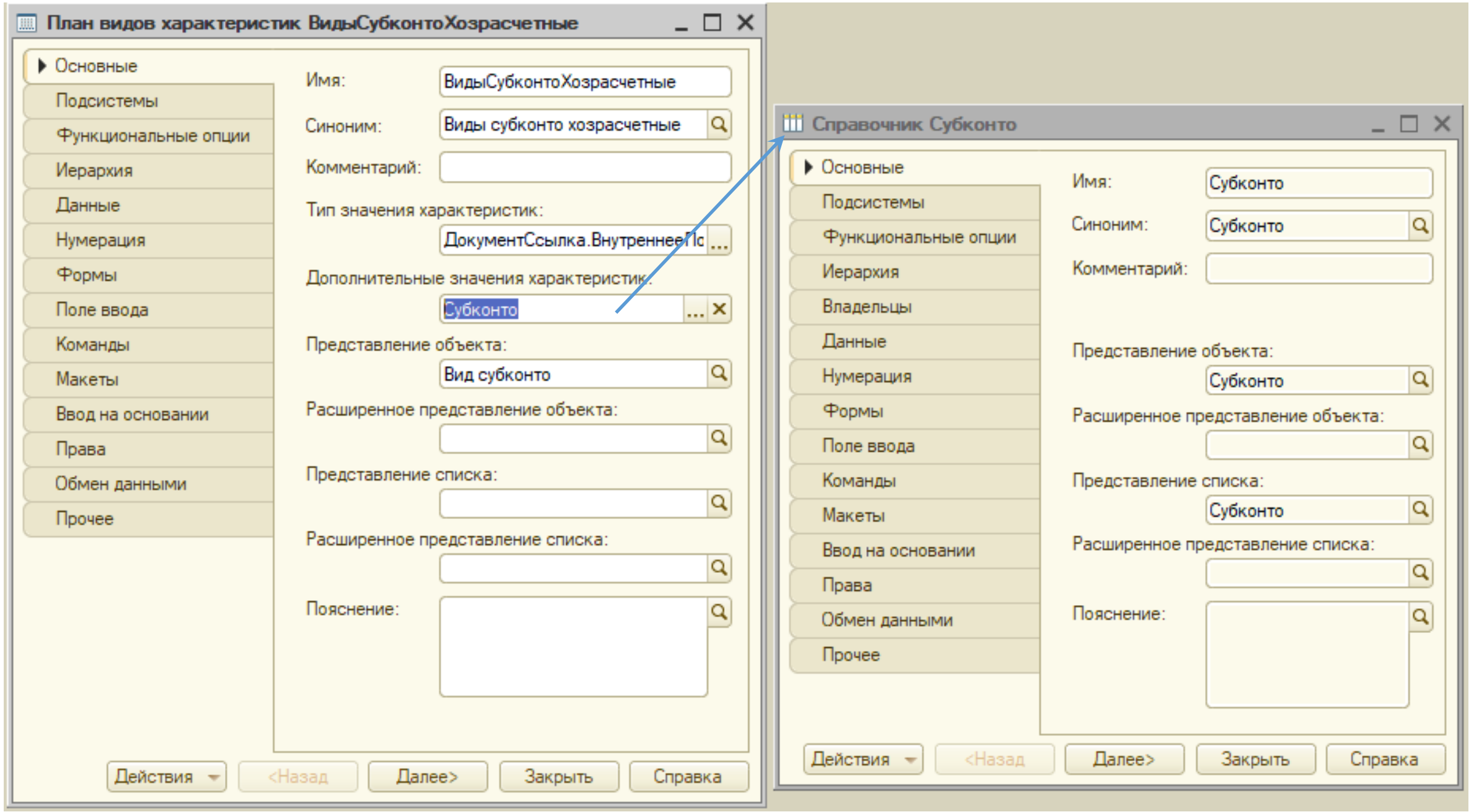
Владельцы справочника «Субконто»







Настройка дополнительных значений характеристик

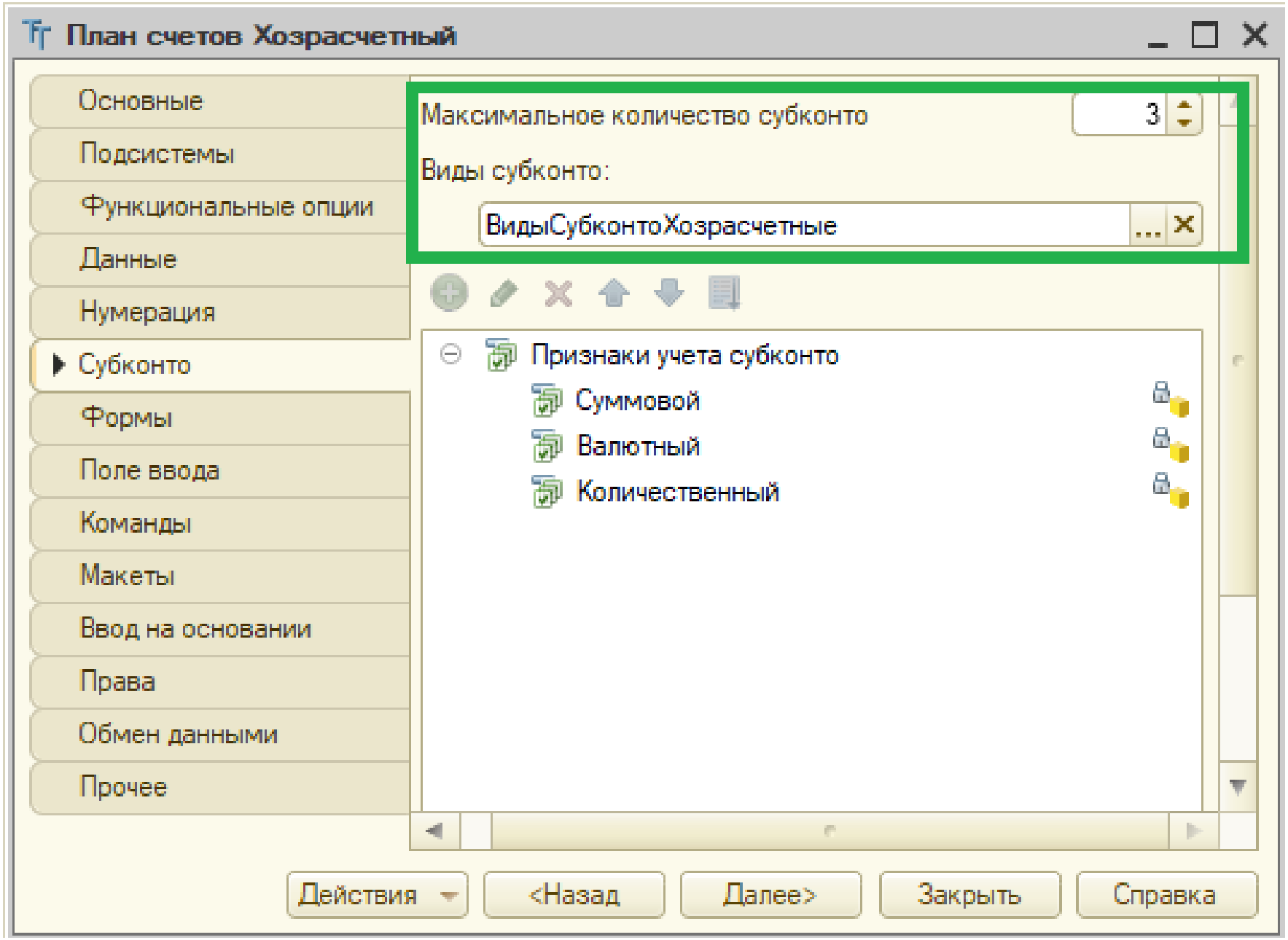


Настройка плана счетов для аналитического учета

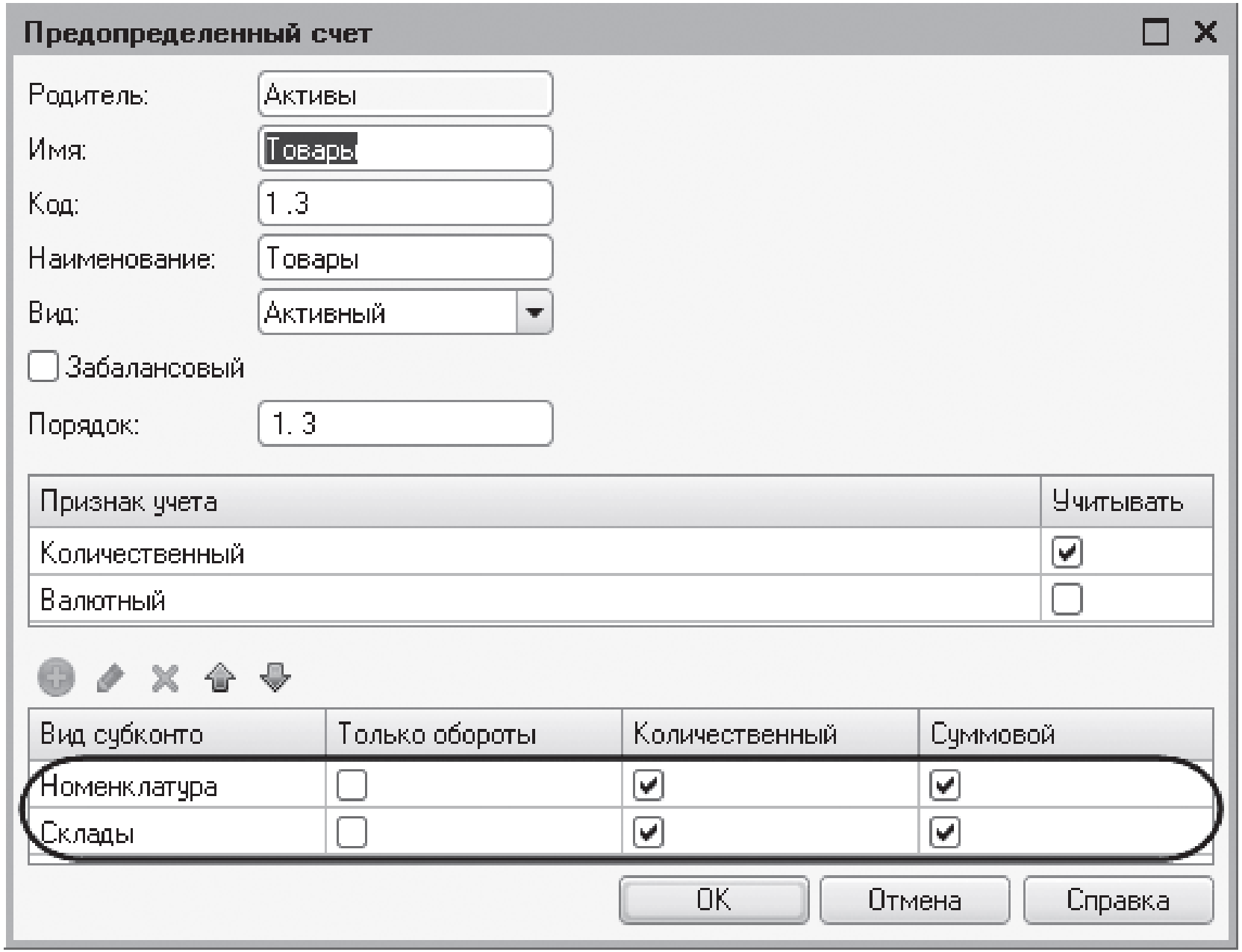
- На закладке Субконто в свойстве Виды субконто выбираем план видов характеристик, который хранит виды субконто для этого плана счетов.

- Определить максимальное количество субконто на счете. Это число параллельных аналитических срезов (видов субконто), которое можно будет выбрать для каждого счета. максимально возможное поддерживаемое платформой **значение 50.**

Настройка плана счетов для аналитического учета



Настройка видов субконто у предопределенного счета



Пример плана счетов

План счетов Хозрасчетный: Предопределенные счета

Действия

Имя	Код	Наименование	Вид	З...	Порядок	Вал
Счета						
Вспомогательный	000	Вспомогательный счет	Активный/Пасс...		00	
ОсновныеСредства	01	Основные средства	Активный		01	
ОСвОрганизации	01.01	Основные средства в ...	Активный		01.01	
АрендованноеИмущество	01.03	Арендованное имущес...	Активный		01.03	
ОСБезГосРегистрации	01.08	Объекты недвижимос...	Активный		01.08	
ВыбытиеОС	01.09	Выбытие основных ср...	Активный		01.09	
КорректировкаСтоимостиАрендованногоИмущества	01.К	Корректировка стоим...	Активный		01.К	
ОСвОрганизации_ЦФ	01.ЦФ	Основные средства в ...	Активный		01.ЦФ	
АмортизацияОсновныхСредств	02	Амортизация основны...	Пассивный		02	
АмортизацияОС_01	02.01	Амортизация основны...	Пассивный		02.01	
АмортизацияОС_03	02.02	Амортизация основны...	Пассивный		02.02	
АмортизацияАрендованногоИмущества	02.03	Амортизация арендов...	Пассивный		02.03	
АмортизацияОС_01_ЦФ	02.ЦФ	Амортизация основны...	Пассивный		02.ЦФ	
ДоходныеВложенияВ_МЦ	03	Доходные вложения в ...	Активный		03	
МЦвОрганизации	03.01	Материальные ценнос...	Активный		03.01	
МЦ_ПредоставленныеВоВременноеВладение	03.02	Материальные ценнос...	Активный		03.02	
МЦ_ПредоставленныеВоВременноеПользование	03.03	Материальные ценнос...	Активный		03.03	
ПрочиеДоходныеВложения	03.04	Прочие доходные вло...	Активный		03.04	
ВыбытиеМЦ	03.09	Выбытие материальн...	Активный		03.09	
НематериальныеАктивы	04	Нематериальные акти...	Активный		04	
НематериальныеАктивыОрганизации	04.01	Нематериальные акти...	Активный		04.01	
РасходыНаНИОКР	04.02	Расходы на научно-исс...	Активный		04.02	
АмортизацияНематериальныхАктивов	05	Амортизация нематер...	Пассивный		05	
ОборудованиеКУстановке	07	Оборудование к устан...	Активный		07	
ВложенияВоВнеоборотныеАктивы	08	Вложения во внеоборо...	Активный		08	
ПриобретениеЗемельныхУчастков	08.01	Приобретение земель...	Активный		08.01	
ПриобретениеОбъектовПриродопользования	08.02	Приобретение объект...	Активный		08.02	
СтроительствоОбъектовОсновныхСредств	08.03	Строительство объект...	Активный		08.03	
ПриобретениеОбъектовОсновныхСредств	08.04	Приобретение объект...	Активный		08.04	
КомпонентыОсновныхСредств	08.04.1	Компоненты основных...	Активный		08.04.1	
ПодготовкаКВводуВЭксплуатацию	08.04.2	Подготовка к вводу в ...	Активный		08.04.2	
ПриобретениеНематериальныхАктивов	08.05	Приобретение немате...	Активный		08.05	
ПереводМолодняка ЖивотныхВОсновноеСтадо	08.06	Перевод молодняка ж...	Активный		08.06	
ПриобретениеВзрослыхЖивотных	08.07	Приобретение взросл...	Активный		08.07	
ВыполнениеНИОКР	08.08	Выполнение научно-ис...	Активный		08.08	
НематериальныеПоисковыеАктивы	08.11	Нематериальные поис...	Активный		08.11	
МатериальныеПоисковыеАктивы	08.12	Материальные поиско...	Активный		08.12	

Пример использования свойства «ВидСубконто»

```
Для Каждого ВидСубконто Из Счет.ВидыСубконто Цикл
        Вид = ВидСубконто.ВидСубконто;
        ТипЗначения = Вид.ТипЗначения;
        Наименование = Вид.Наименование;
        ВведенВКонфигураторе =
                                ВидСубконто.Предопределенное;
        Оборотный = ВидСубконто.ТолькоОбороты;
        Количественный = ВидСубконто.Количественный;
КонецЦикла;
```

От организации плана счетов и аналитического учета во многом зависит не только функциональность, но и производительность конфигурации

- Задача. Пользователь изъявил желание вести учет товаров в разрезе номенклатуры товаров. Список товаров существенный и незакрытый (не добавляются новые позиции, удаляются старые). Кроме кода и наименования номенклатуры необходимо хранить и другую дополнительную, но важную информацию.
- Ниже приводится таблица, позволяющая в конкретном случае принять решение об оптимальности использования механизма платформы.

Решение по выбору субсчета или субконто

Критерий	Учет на субсчетах	Учет на субконто
Количество позиций	От единиц до десятков	От десятков до бесконечности
Список позиций учета закрыт	Как правило, да	Может и будет расширяться
Параллельные срезы возможны	Только иерархический учет	Несколько параллельных срезов
Достаточная информативность	Кода и Наименования достаточно	Есть дополнительные свойства

Решение по выбору: новый вид субконто или новое свойство старого

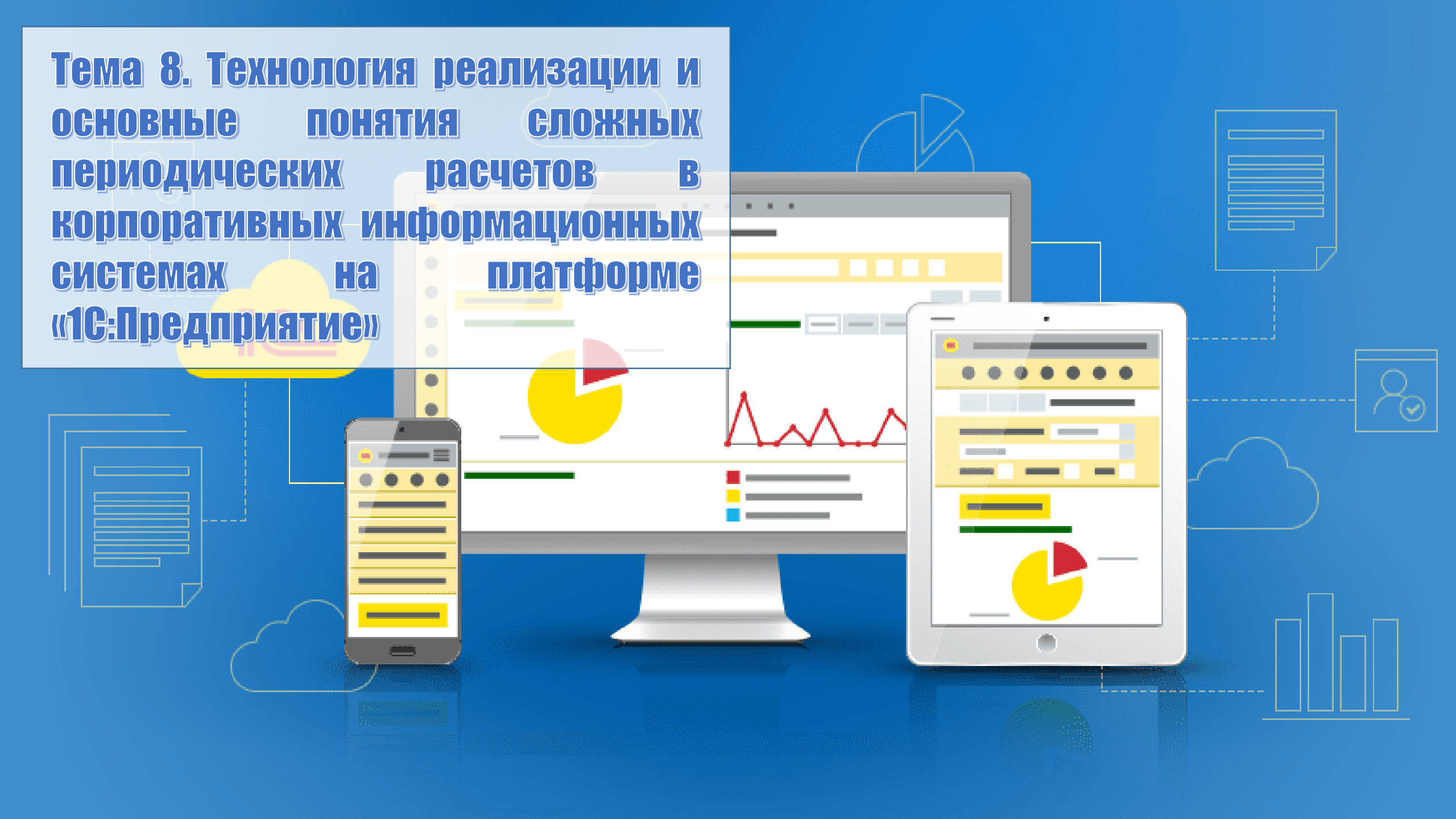
Критерий	**Новый вид субконто**	**Новое свойство субконто**
Можно ли однозначно связать новый критерий отбора (группировки) с существующим объектом аналитического учета	Связать невозможно, это параллельный срез	Новый критерий вполне можно разместить как новое свойство существующего вида аналитики

Решение по выбору порядка следования видов субконто

Критерий	Субконто бухгалтерии	Измерения оперативного учета
Максимальное количество субконто, выбранное в плане счетов демонстрационной конфигурации	Не более двух	Нет ограничения, может быть создан регистр с нужным количеством измерений
Максимальное количество субконто, с которыми эффективно может работать платформа	* Не представляется возможным дать этот критерий в количественном выражении, не увидев и не проанализировав задачу, которую необходимо решить. Можно сказать лишь, что при решении одной и той же задачи автоматизации одного раздела учета наличия и движения средств регистр накопления будет эффективнее регистра бухгалтерии. Это обусловлено универсальностью регистра бухгалтерии, платой за которую является эффективность.	
Количество объектов учета	Десятки, сотни, тысячи **	Десятки и сотни тысяч **
Количество документов в день	Единицы, десятки **	Десятки, сотни **
Измерение примитивного типа	Настоятельно не рекомендуется	В принципе возможно
Влияние на бухгалтерский баланс	Необходимо	Не рекомендуется

** Количественные оценки очень приблизительны и неточны, и причина этого все та же: невозможно дать универсальные рекомендации. Правильное решение должно быть результатом оценки каждой конкретной задачи, и осуществляется эта оценка на этапе технического проектирования задачи после детального изучения объекта автоматизации.

Тема 8. Технология реализации и основные понятия сложных периодических расчетов в корпоративных информационных системах на платформе «1С:Предприятие»



- *Технология реализации и основные понятия сложных периодических расчетов в корпоративных информационных системах на платформе «1С:Педприятие»:*
  - *вид расчета,*
  - *период регистрации,*
  - *период действия,*
  - *вытесняющие расчеты и фактический период действия,*
  - *зависимость по базовому периоду,*
  - *ведущие расчеты и перерасчет,*
  - *сторнирование.*

Работа механизма сложных периодических расчетов

Особенности периодических расчетов:

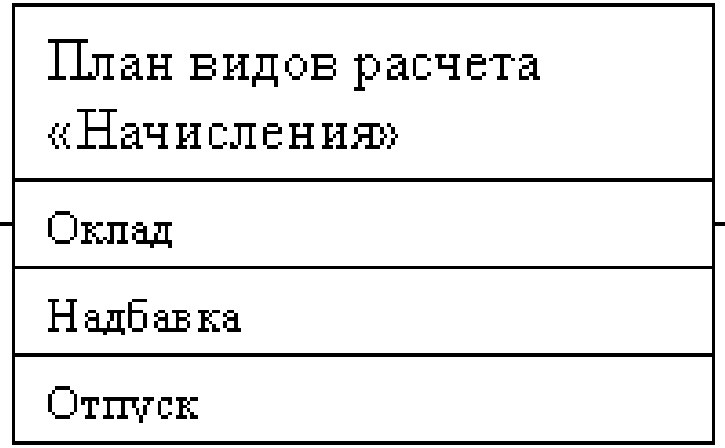
- отсутствие однозначной привязки событий к точке на оси времени. Регистрируемые события в этом виде учета имеют отношение не к моменту времени, а к периоду в целом. ;
- протяженность некоторых регистрируемых событий во времени.

Основные понятия периодических расчетов

Вид расчета: Любой расчет, выполняемый в системе, регистрируется с обязательным указанием вида расчета, под которым может пониматься как способ расчета данной записи, так и дополнительные свойства, характеризующие сущность именно этого расчета.

Период. Для расчетов важным является понятие периода. Обычно период описывается датой начала и датой окончания. Если для расчета определена периодичность (см. описание ниже), то для описания периода (действия, регистрации) данного расчета достаточно указать любую дату. По этой дате вычисляется дата начала периода, и именно эта дата будет описывать период. Такой порядок определения периода позволяет оптимизировать выполнение запросов, в которых требуется выбрать записи, относящиеся к указанному периоду.

Использование видов расчета в разных регистрах расчета



Регистр расчета «Управленческие начисления»

Период регистрации	Вид расчета	Сотрудник	Сумма, USD
Март 2021	Оклад	Иванов	400
Март 2021	Надбавка	Иванов	20
Апрель 2021	Оклад	Петров	300
…	…	…	…

Регистр расчета «Регламентированные начисления»

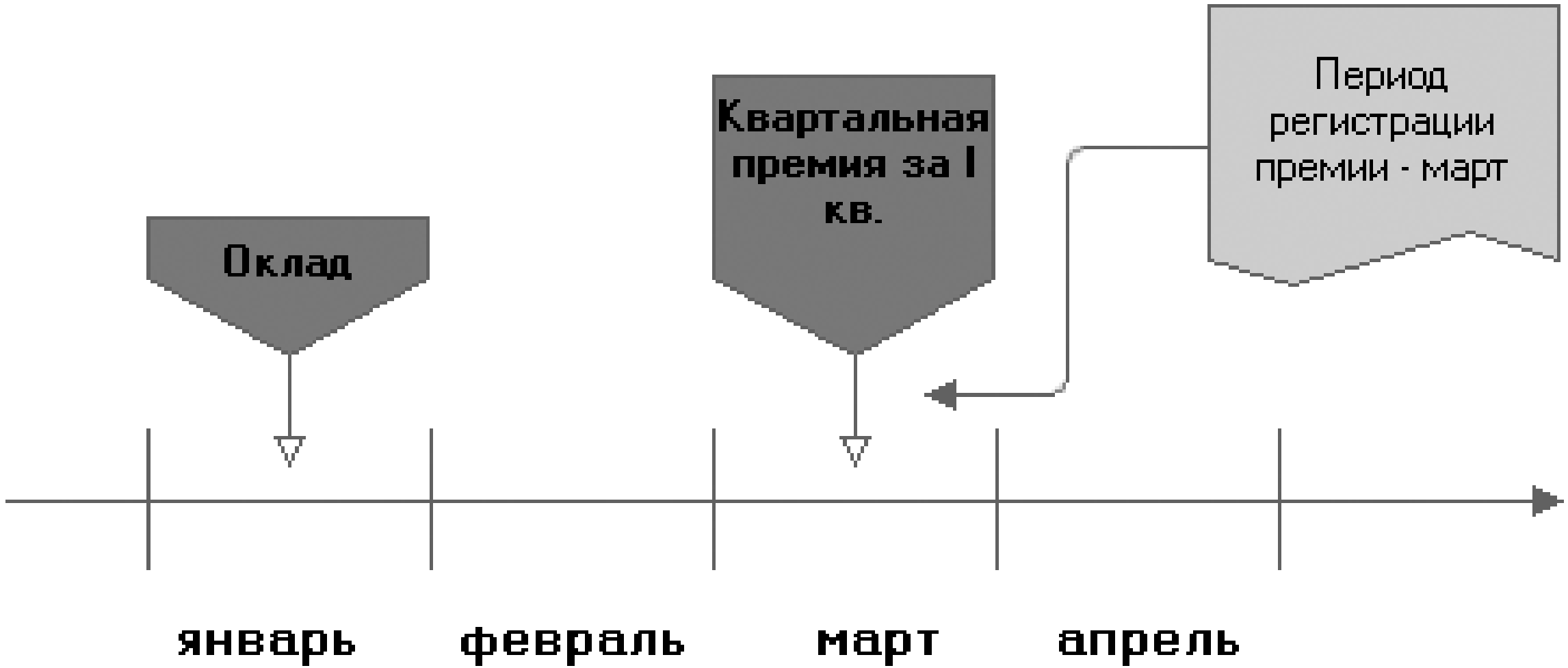
Период регистрации	Вид расчета	Сотрудник	Сумма, руб.
Март 2021	Оклад	Сидоров	8 000
Апрель 2021	Оклад	Сидоров	7 200
Апрель 2021	Отпуск	Иванов	6 000
…	…	…	…

Периодичность расчетов. Определяет, с каким периодом будут (могут) выполняться расчеты, учитываемые данным регистром. Задается в свойстве Периодичность регистра расчетов. По значению этого свойства (если регистр периодический) определяется период действия записи регистра расчета. Например, регистр имеет периодичность Месяц, тогда при формировании записи регистра в качестве периода действия выбирается дата документа, и по ней система определяет период действия.

Период действия - начальная дата периода, определяемая в соответствии со значением свойства Периодичность.

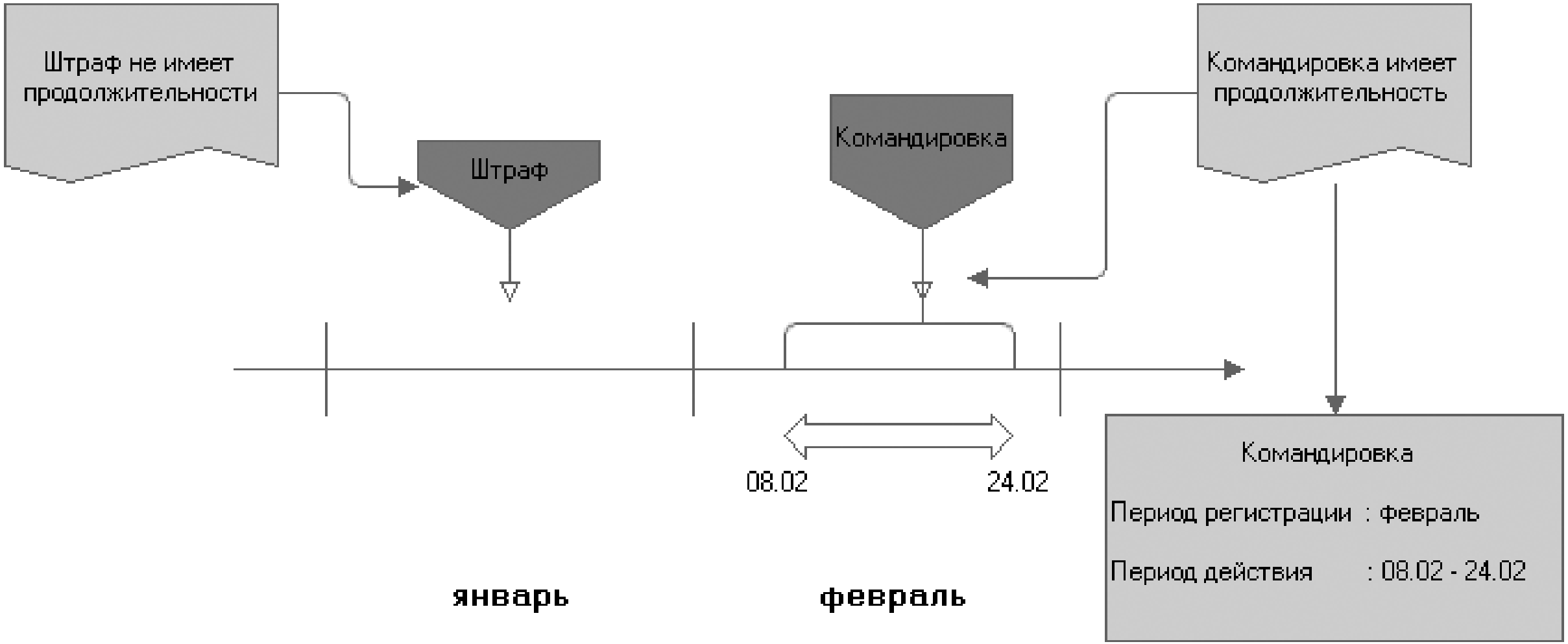
Период действия расчета - указывает период, за который производится расчет. Период определяется датой начала и датой окончания периода

Привязка записей расчета к периоду регистрации

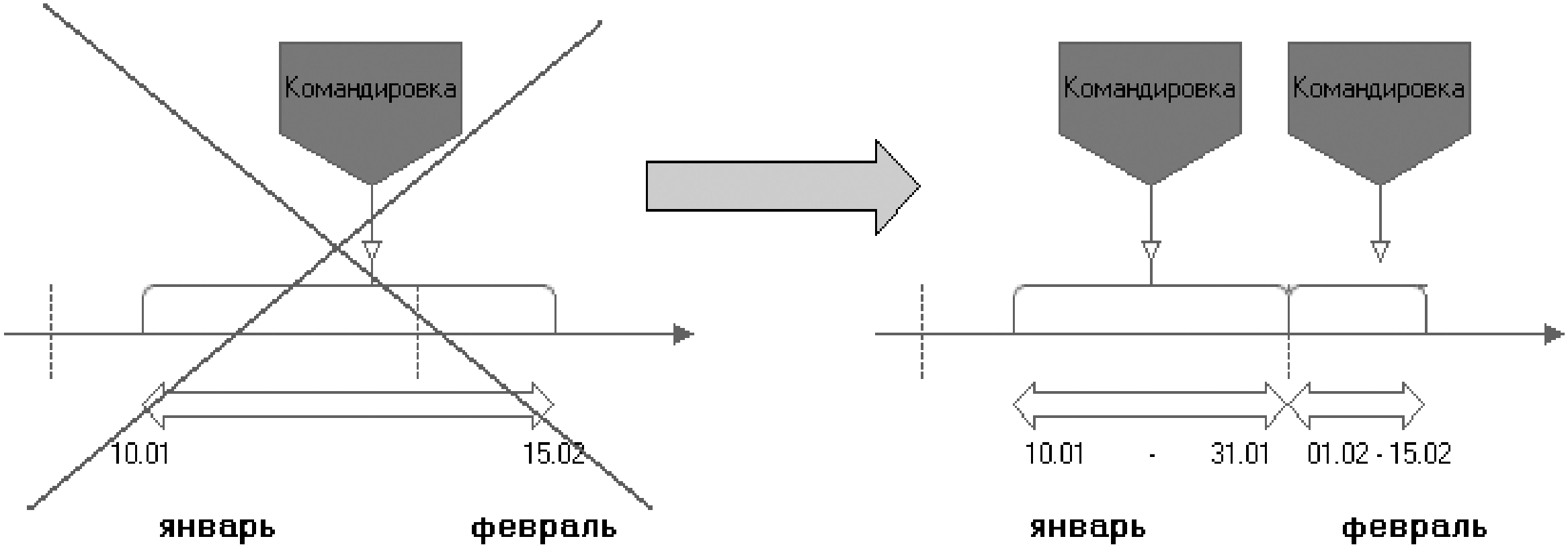


Периодичность расчетов: месяц

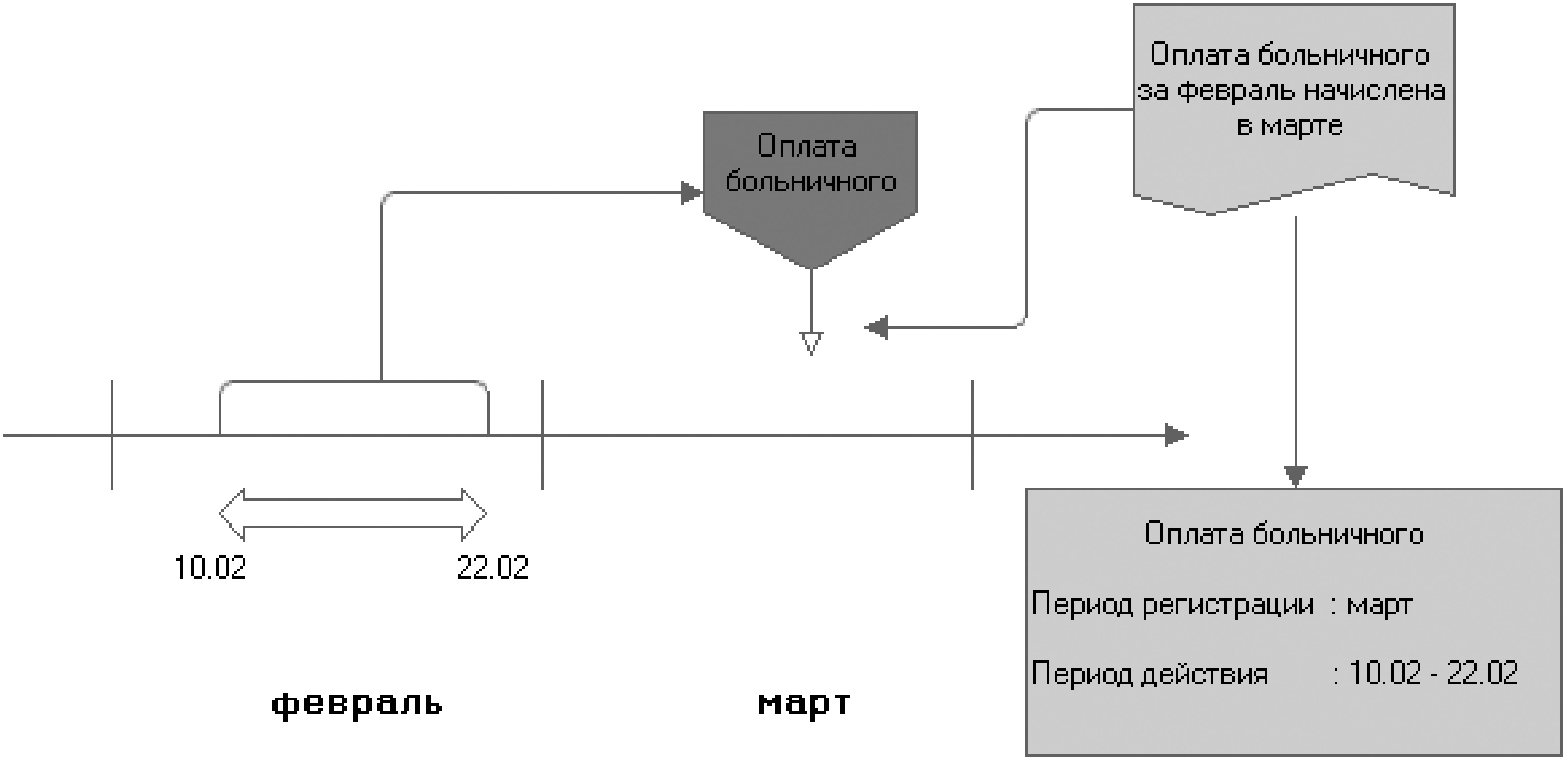
Записи расчета, протяженные во времени



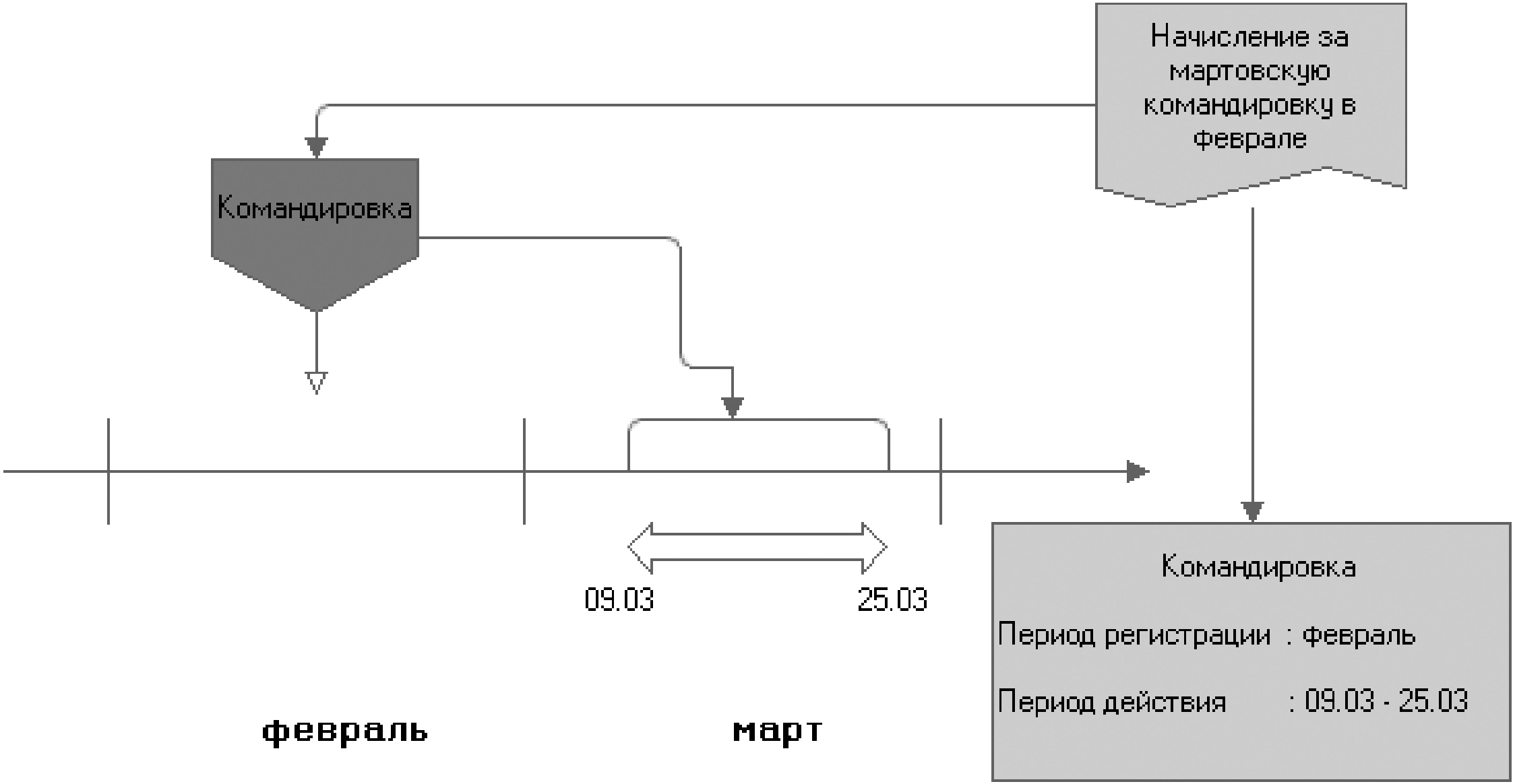
Ввод расчета, который длится в нескольких периодах



Период действия раньше периода регистрации



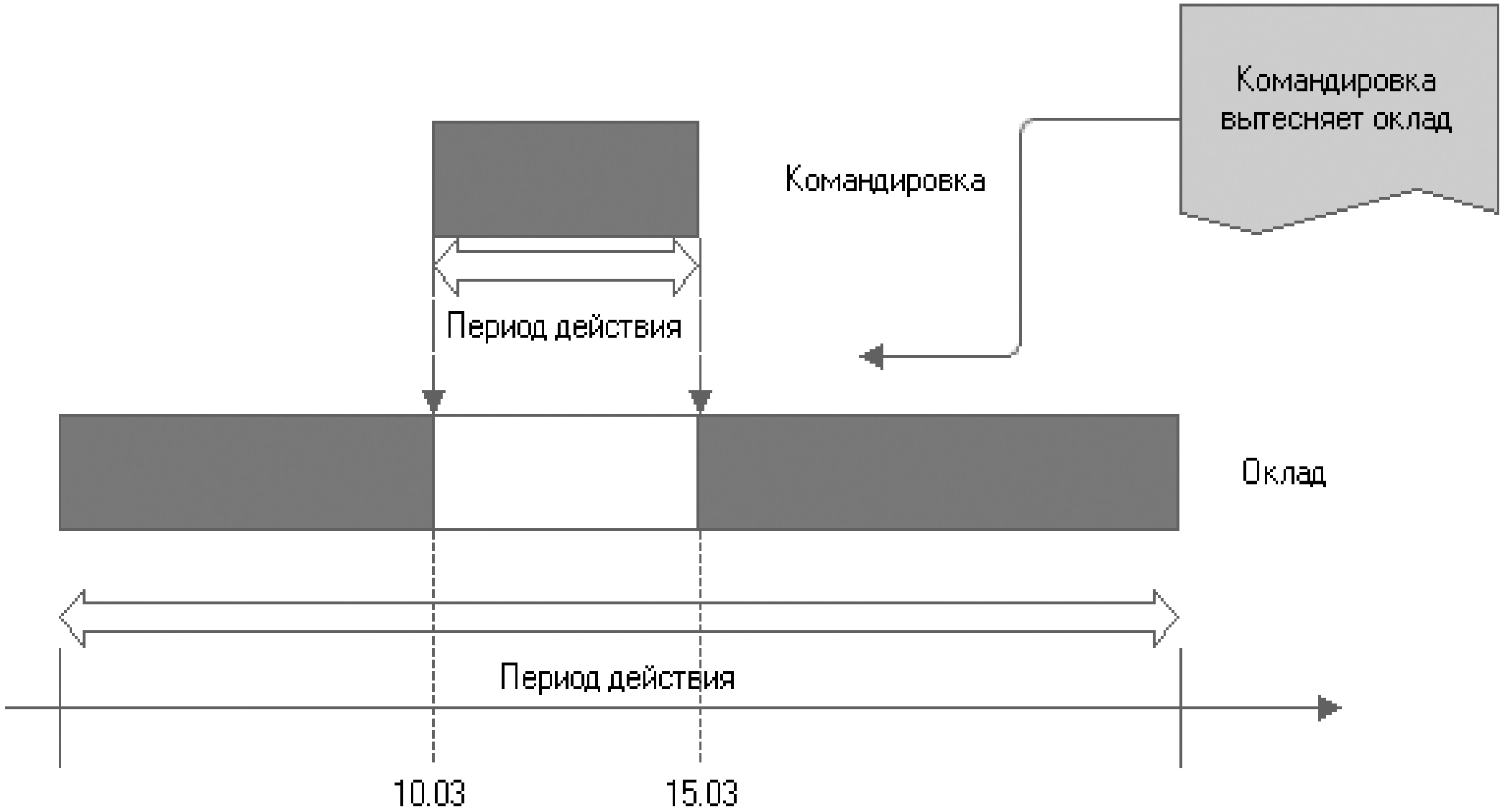
Период действия позже периода регистрации



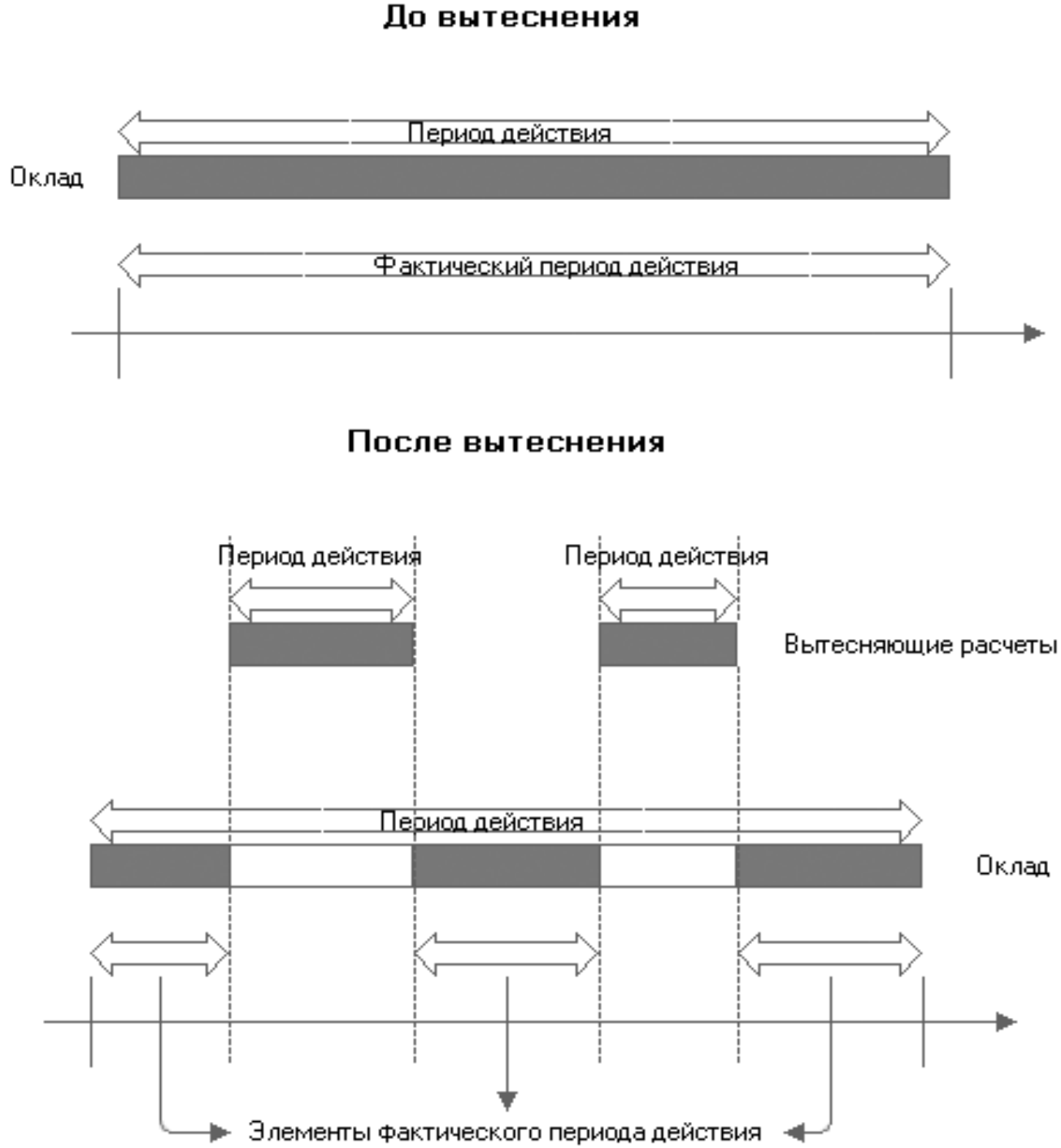
Механизм вытеснения

- **Вытесняющие расчеты и фактический период действия**. Протяженные во времени записи регистров расчета могут конкурировать между собой за период действия. Это означает, что они не могут действовать одновременно. Такая конкуренция необходима, когда виды расчета являются по сути взаимоисключающими.
- Например, виды расчета *Оклад* и *Командировка* не могут действовать в один и тот же момент, так как сотрудник не может одновременно работать на основном месте и находиться в командировке. При вводе записи о командировке на определенный интервал времени система должна исключать действие оклада в этом интервале. Такие виды расчетов, которые исключают одновременное действие записей других видов расчетов, называются вытесняющими

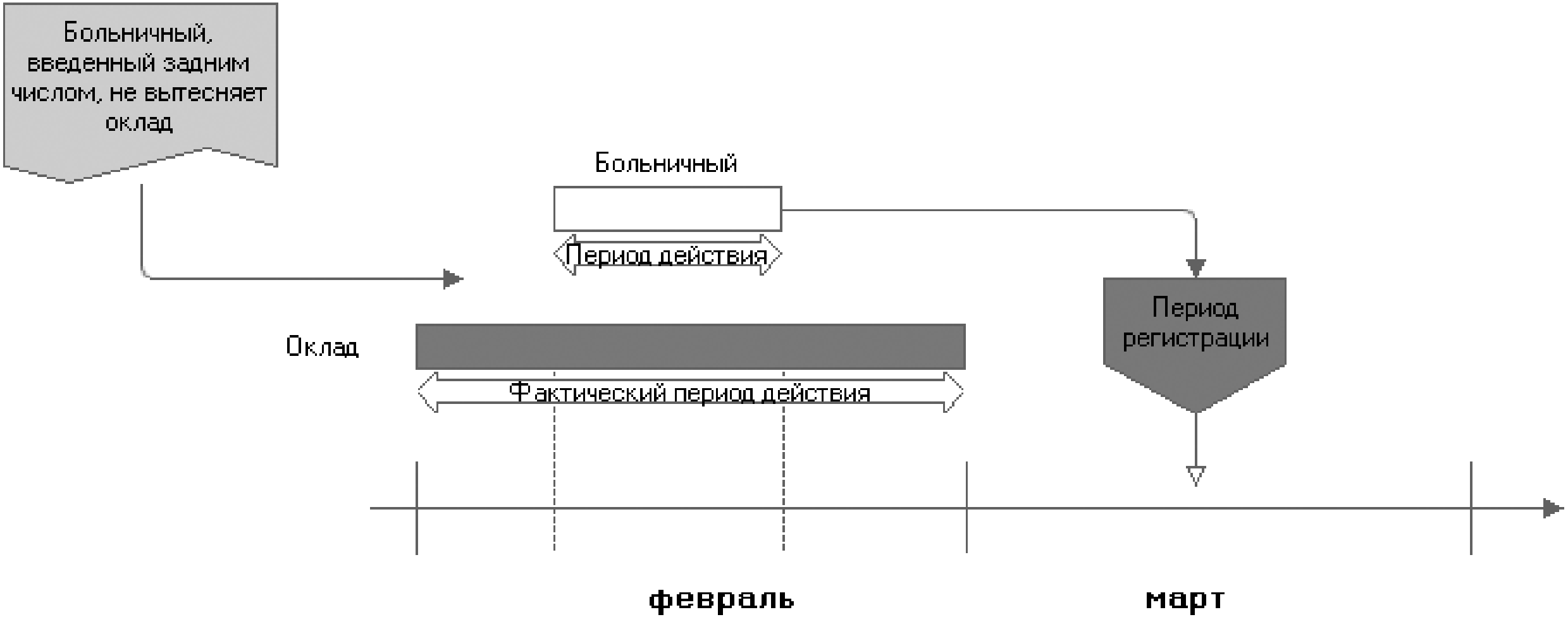
Вытеснение по периоду действия



Фактический период действия



Записи ранних периодов не вытесняются

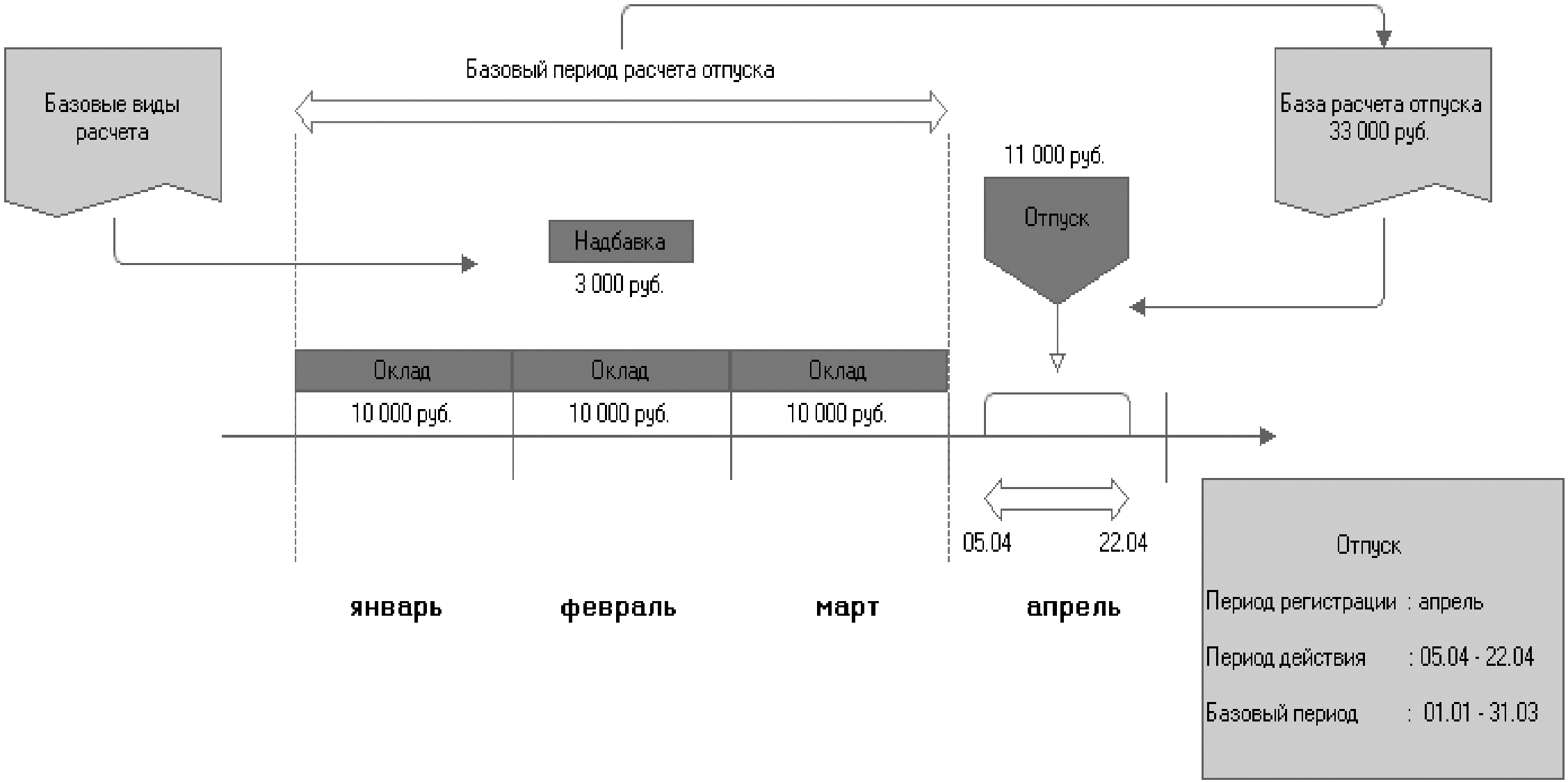


Фактический период действия – если расчет не вытесняется другими расчетами, то фактический период совпадает с периодом действия. Если есть вытесняющие виды расчетов, то фактический период определяется как совокупность непересекающихся периодов, в которых данный расчет не вытеснялся.

Зависимость по базовому периоду. В определенных случаях результат одного расчета может зависеть от результата других расчетов, введенных в систему. Например, квартальная премия может зависеть от суммы начисленного за квартал заработка по окладу. В этом случае говорят, что оклад входит в базу расчета премии, а вид расчета Оклад является базовым по отношению к виду расчета Квартальная премия. При этом один вид расчета может иметь несколько базовых видов расчета, то есть зависеть одновременно от нескольких других расчетов. При расчете результата расчета по базе анализируется сумма базовых видов расчета за определенный период, который называется базовым периодом. Базовый период – это произвольный непрерывный интервал дат, который может покрывать несколько расчетных периодов. Например, при расчете оплачиваемого отпуска, как правило, производится расчет среднего заработка за 3 месяца, предшествующих отпуску. Этот интервал и будет базовым периодом расчета отпуска.

Базовый период – определяет период, за который будут выбираться результаты расчетов, используемых (являющихся базовыми) для данного расчета

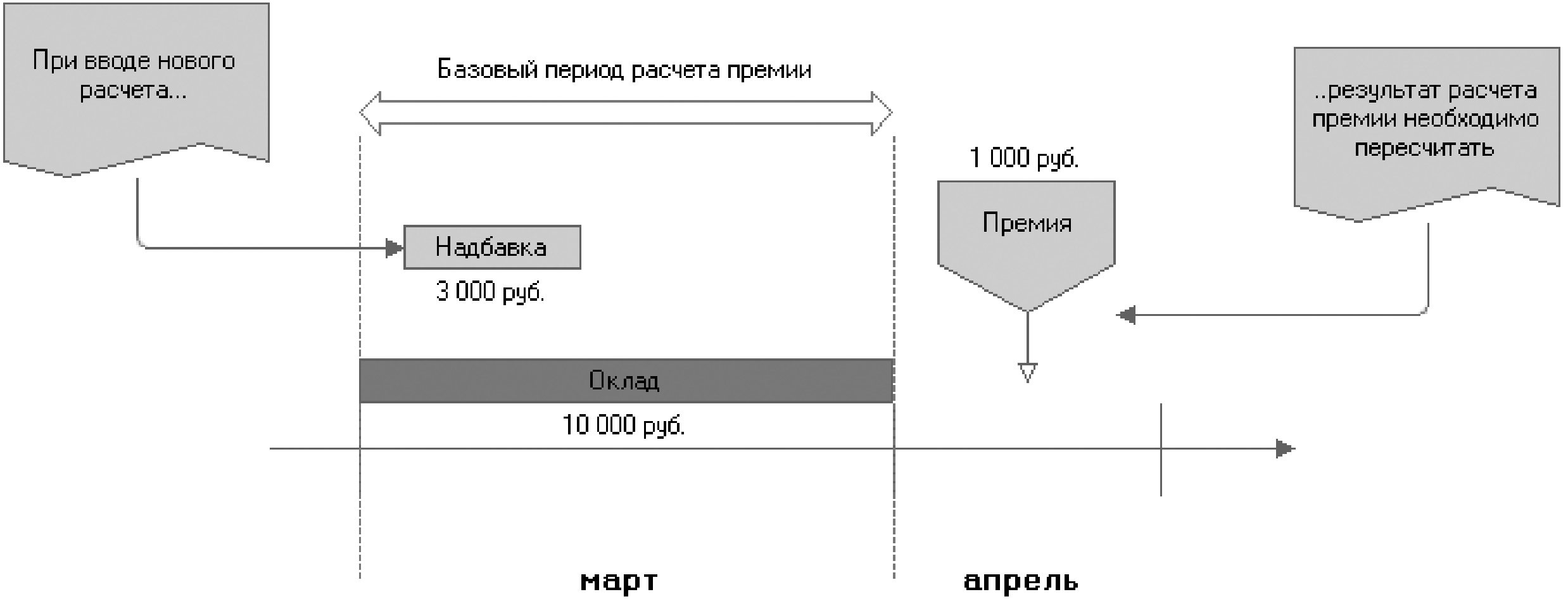
Базовый период



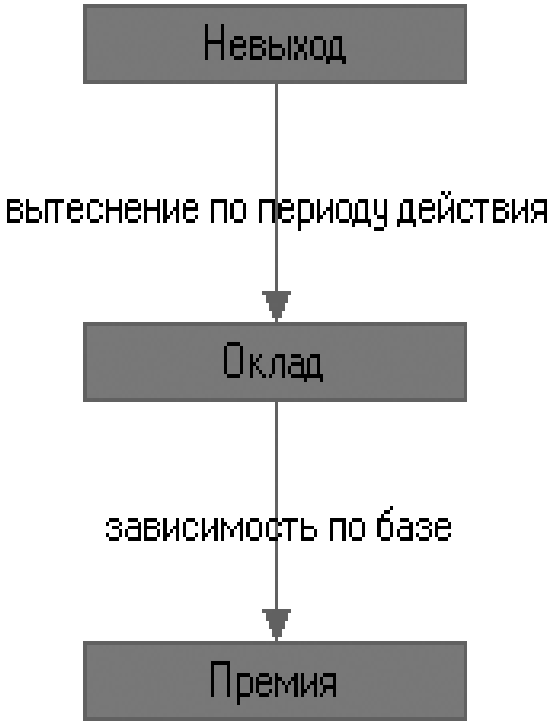
Ведущие виды расчета и перерасчет

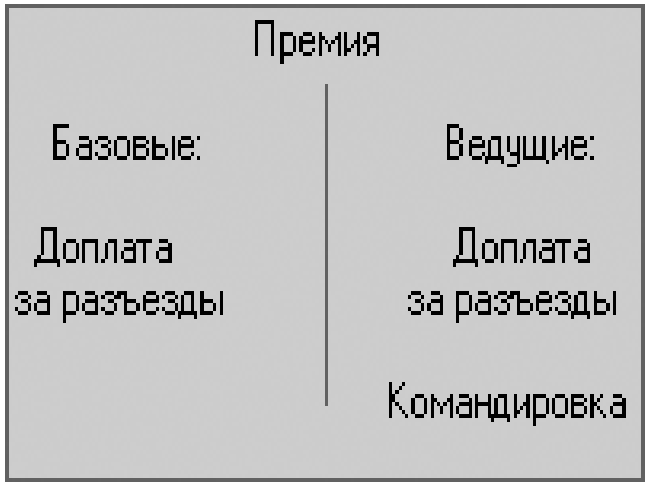
- **Ведущие расчеты и перерасчет**. Ведущими называют виды расчетов, при вводе или изменении которых необходимо перерассчитать результат уже существующих расчетов. Например, если работнику начислена премия за март в размере 10 % от заработка, то при вводе нового начисления в марте размер этой премии теряет актуальность. Премию необходимо пересчитать, чтобы учесть новое начисление.

Необходимость перерасчета при изменении базового вида расчета



Ведущие виды расчета

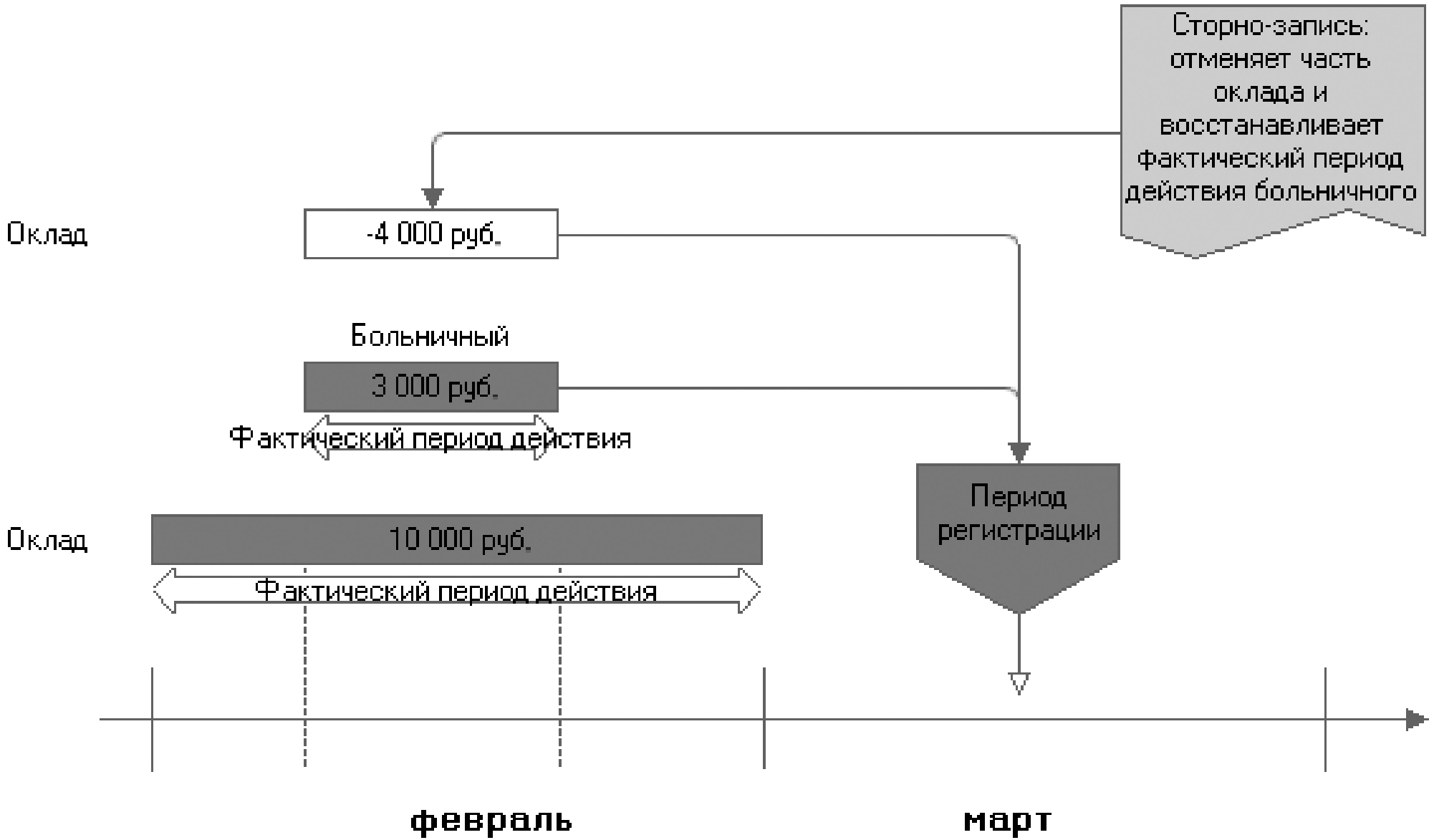




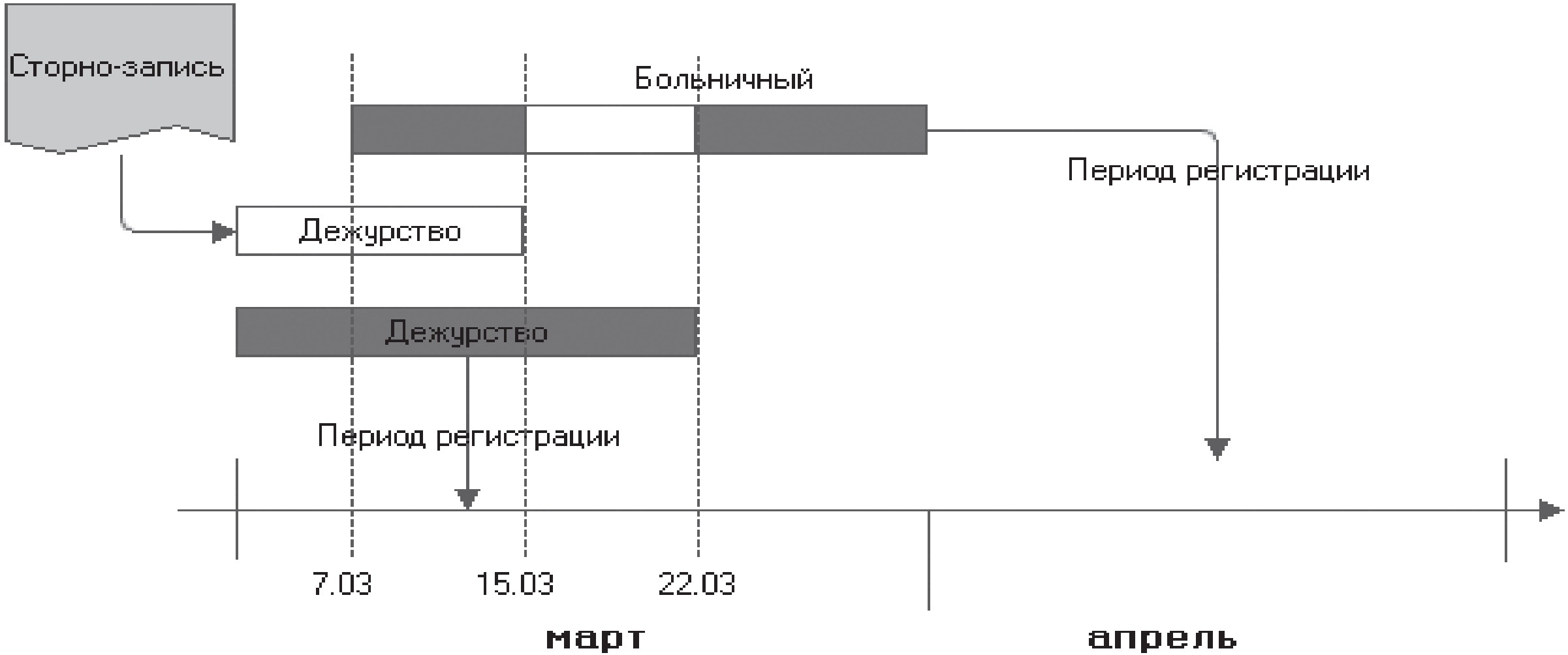
Сторнирование

- **Сторнирование.** Возможность указать для записи период действия, отличный от периода регистрации, существует только у регистров, поддерживающих период действия. В этом случае, если у записи период действия раньше периода регистрации (то есть запись зарегистрирована «с опозданием»), эта запись может вступать в конкуренцию за период действия с записями более раннего периода регистрации. При этом механизм вытеснения действовать не будет, так как запись имеет более поздний период регистрации, но конкуренция видов расчета останется. В результате у записи с более поздним периодом регистрации может образоваться меньший фактический период действия в силу того, что она не может действовать одновременно с конкурирующими видами расчета

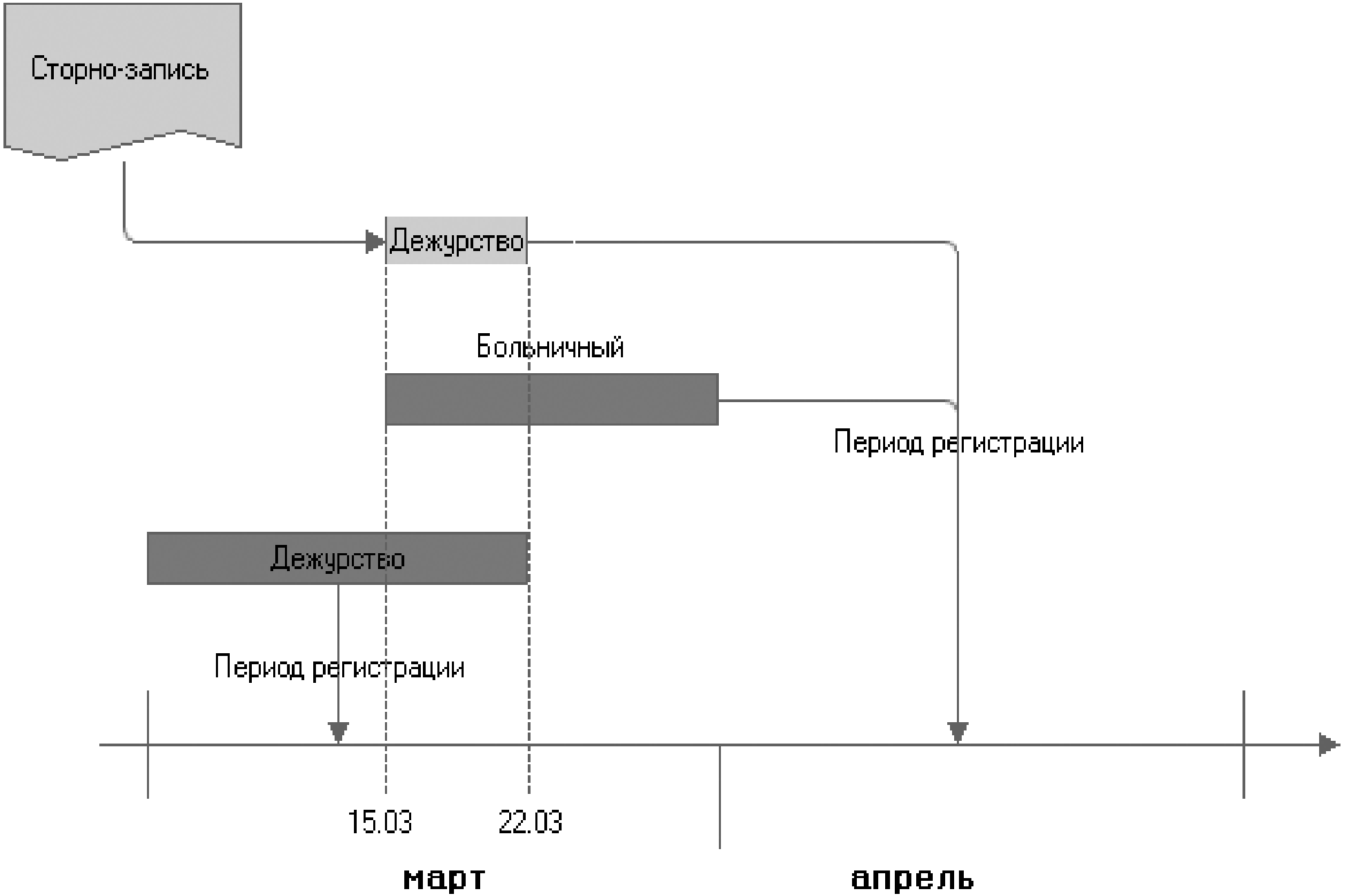
Сторно-записи



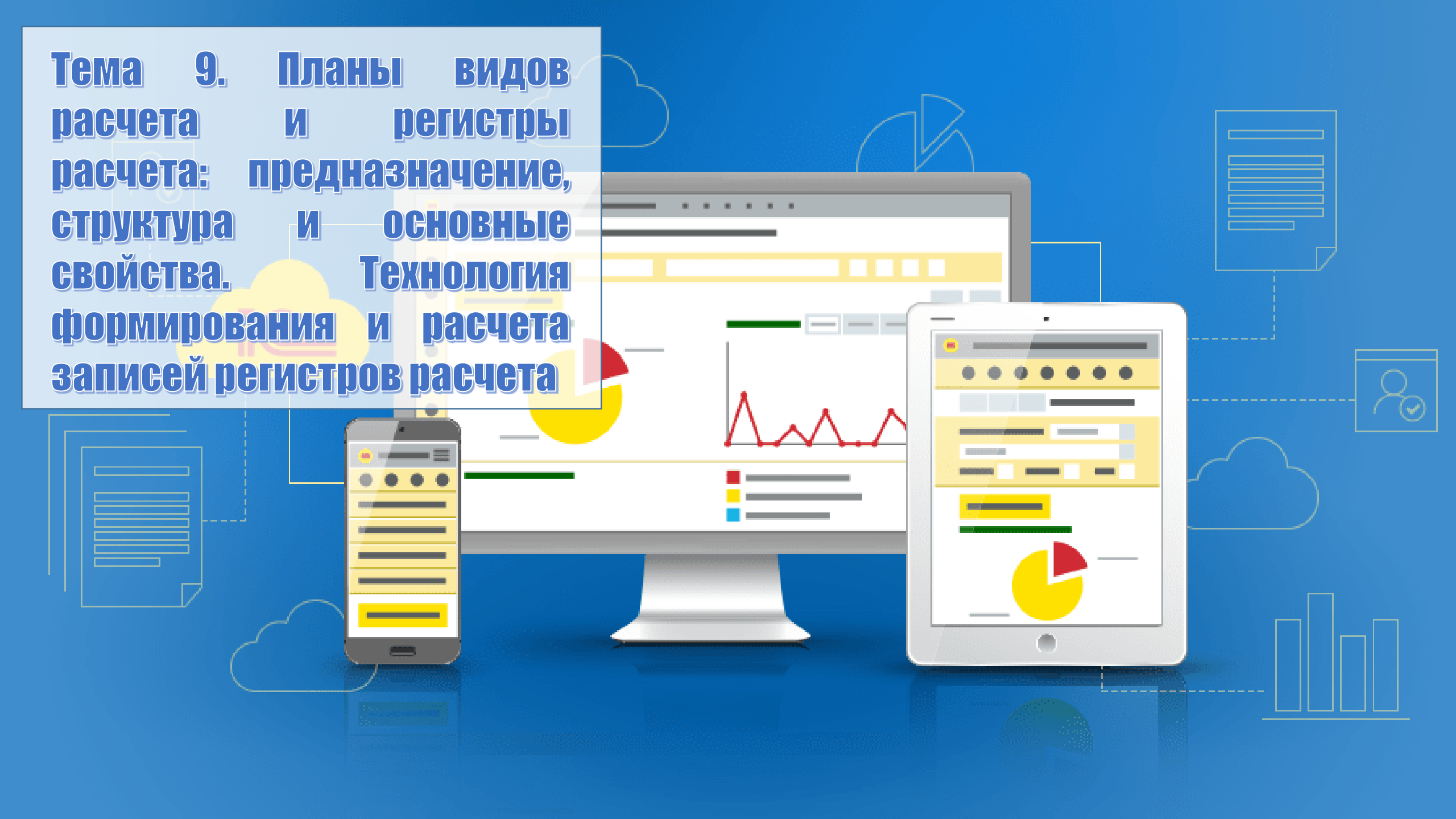
Учет сторно-записей при формировании фактического периода действия



Использование сторно-записей



Тема 9. Планы видов расчета и регистры расчета: предназначение, структура и основные свойства. Технология формирования и расчета записей регистров расчета



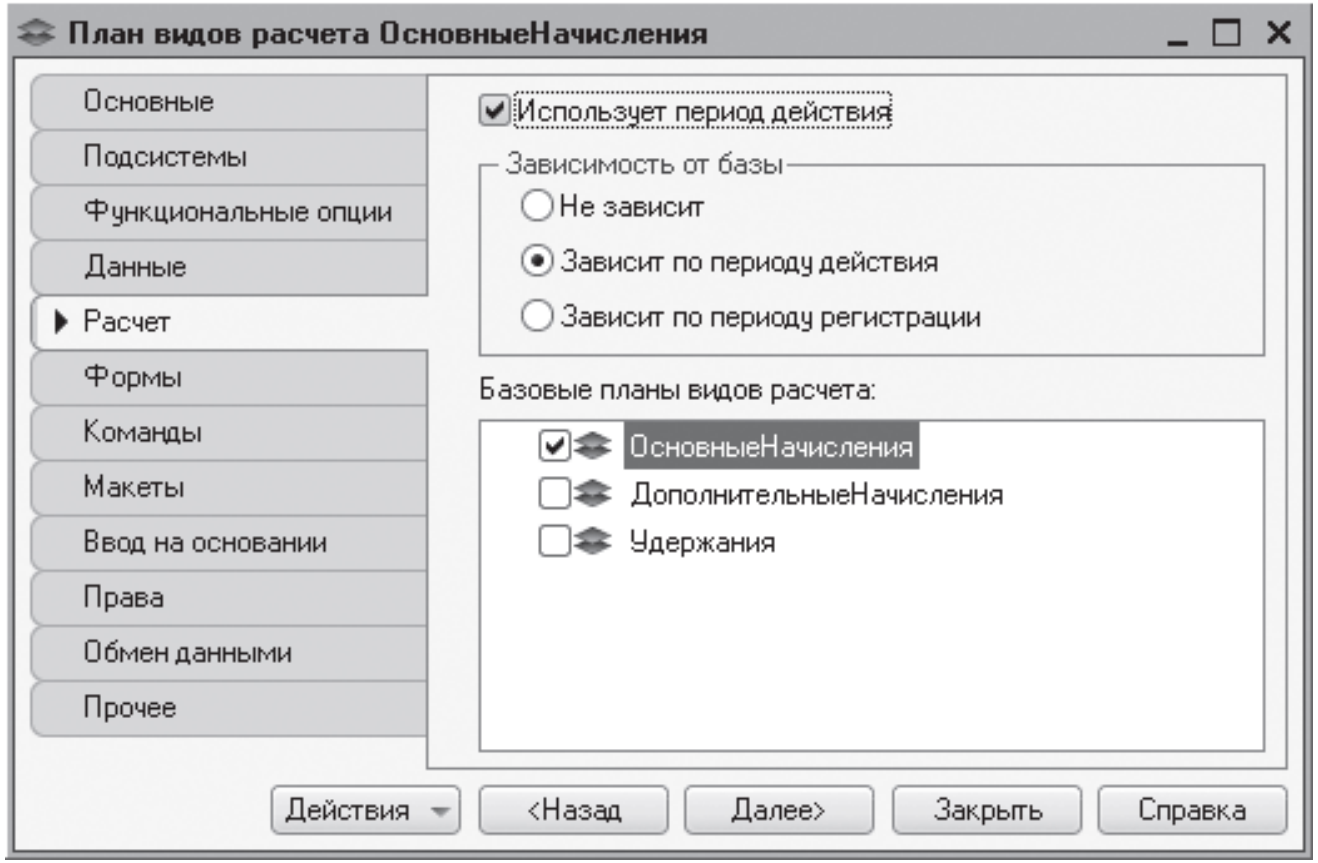
- *Планы видов расчета и регистры расчета.*
- *Назначение, структура и свойства планов видов расчета.*
- *Назначение, структура и свойства регистра расчета.*
- *Настройка протяженных по времени расчетов: использование механизма вытеснения, использование графиков.*
- *Настройка зависимости по базовому периоду. Технология формирования и расчета записей регистров расчета.*

Планы видов расчета:

- представляют собой прикладные объекты конфигурации, предназначенные для описания структур данных, в которых хранятся однотипные виды расчета;
- в конфигурации может быть создано неограниченное количество планов видов расчета. Планы видов расчета могут различаться между собой по свойствам или по назначению использования;
- При создании и настройке плана видов расчета разработчик может влиять на его свойства. Для каждого плана видов расчета задается его ***Имя***, по которому можно обращаться к этому объекту конфигурации, и Синоним, а также ***Представление объекта***, ***Представление списка*** и т. п. для представления плана видов расчета в интерфейсе «1С:Предприятия».

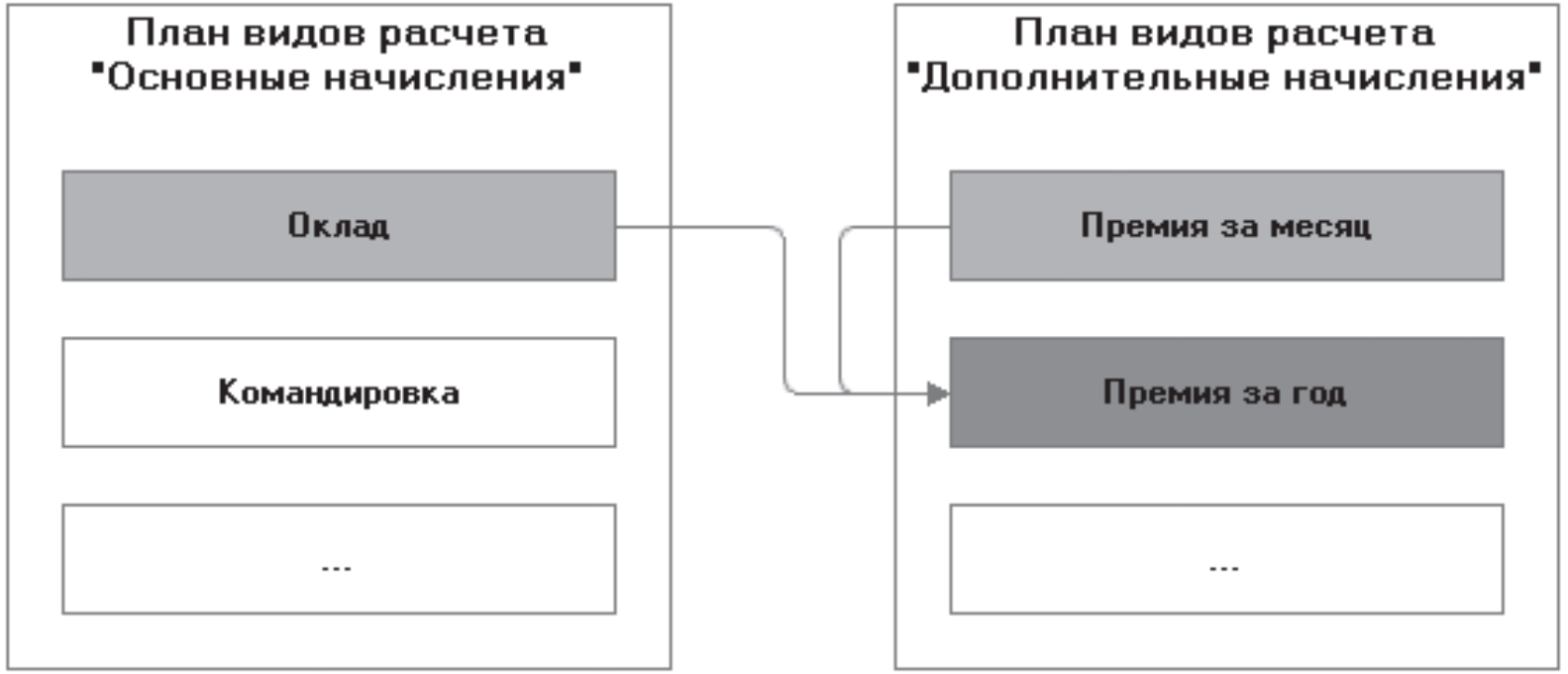
Свойства плана видов расчета

- *Использует период действия*. Если в плане видов расчета установлено свойство «Использует период действия», то все виды расчета, хранящиеся в данном плане, будут рассматриваться как протяженные во времени. Для таких видов расчета применима настройка вытеснения по периоду действия. Планы видов расчета, использующие период действия, можно использовать в регистрах расчета с периодом действия. Если свойство не установлено, все виды расчета данного плана видов расчета будут рассматриваться как непротяженные во времени, для них нет смысла настраивать вытеснение. Такие планы видов расчета не могут быть использованы в регистрах расчета с периодом действия.



Свойства плана видов расчета

- *Зависимость от базы.* Эта настройка определяет, будет ли в видах расчета данного плана видов расчета использоваться зависимость по базовому периоду. Если переключатель установлен в положение Не зависит, то виды расчета данного плана видов расчета не смогут зависеть по базовому периоду от других видов расчета.



- Установка переключателя в положение Зависит по периоду действия или Зависит по периоду регистрации позволит устанавливать в видах расчета зависимость по базовому периоду. При этом любой вид расчета данного плана видов расчета теоретически может зависеть по базовому периоду от любых других видов расчета, в том числе из других планов видов расчета. Поэтому при настройке зависимости от базы необходимо указать, какие виды расчета могут выступать базовыми для видов расчета данного плана. При этом базовые виды расчета могут принадлежать данному плану счетов или другому плану счетов.

Структура плана видов расчета

Регистры видов расчета

- Это прикладные объекты конфигурации, которые предназначены для учета результатов вычислений, осуществляемых с некоторой периодичностью, тесно связанных друг с другом по некоторым правилам и взаимно влияющих друг на друга в пределах определенного периода (например, расчет заработной платы, расчет квартплаты, расчет арендной платы).
- В конфигурации их может быть неограниченное количество.
- Они так же как и другие регистры имеют измерения, ресурсы, реквизиты, однако, принцип действия регистров расчета абсолютно другой.
- Следует отметить, что взаимозависимость видов расчета настраивается в планах видов расчета, однако, расчетные механизмы платформы заложены именно в регистры расчета.
- Регистры расчета обеспечивают и регистрацию записей, и правильное отображение взаимосвязей расчетов (расчет записи по базе, вытеснение записей, формирование сторно-записи, учет протяженных во времени расчетов, хранение периода действия и фактического периода действия записей). Для расчетов, зависящих от базового периода, регистр расчета также хранит данные о базовом периоде.

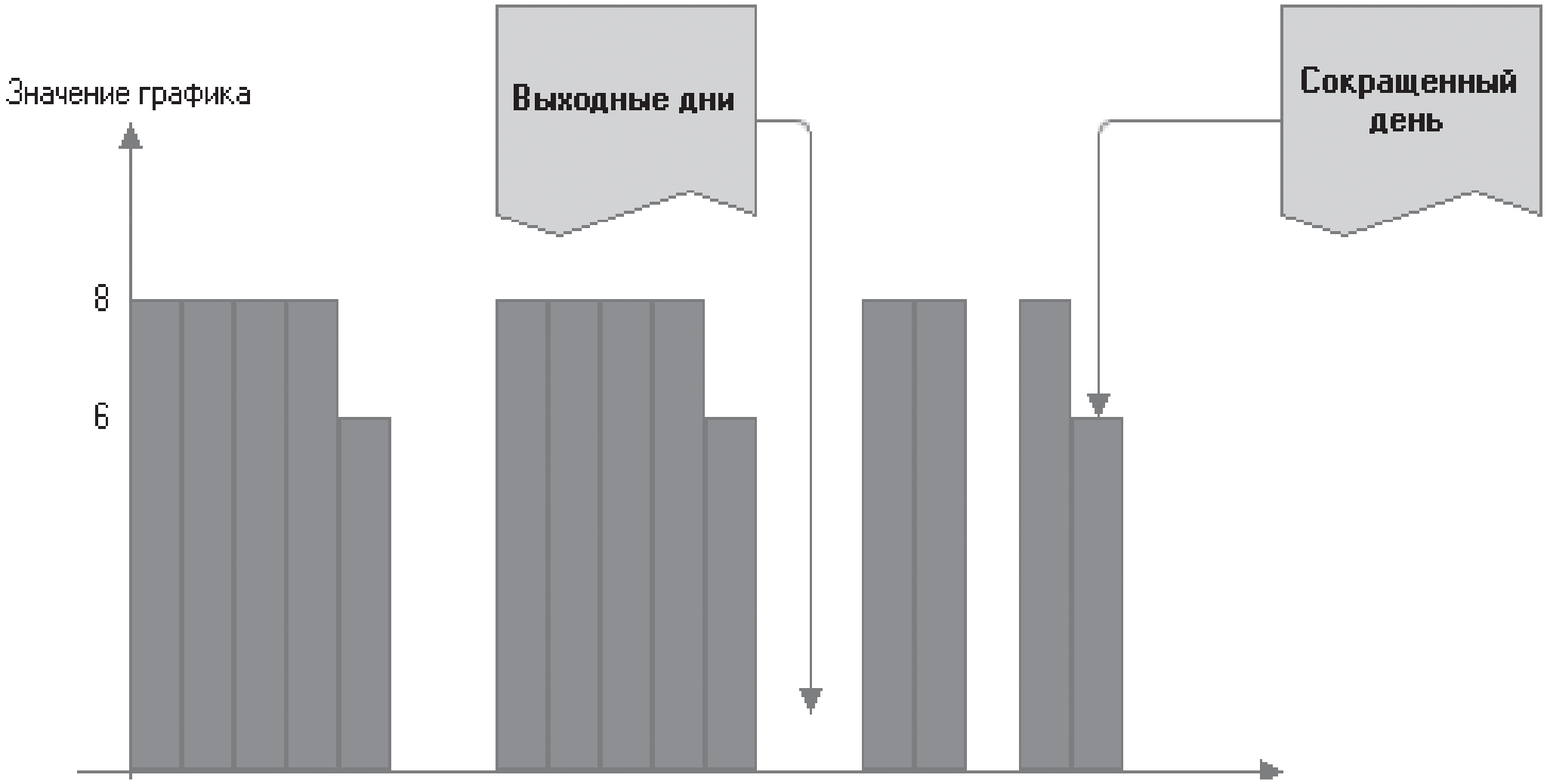
Периодичность регистра расчетов

- Свойство Периодичность регистра расчетов определяет размерность периодов, для которых будет вестись учет в этом регистре.
- При настройке регистра доступны следующие виды периодичности: Год, Квартал, Месяц и День. Соответственно, записи в таком регистре будут фиксироваться в привязке к конкретному году, кварталу, месяцу или дню.
- Периодичность регистра определяет вид периода регистрации записей и зависит от задачи, которую он решает. Для целей расчета зарплаты, как правило, применяют регистры с периодичностью Месяц.

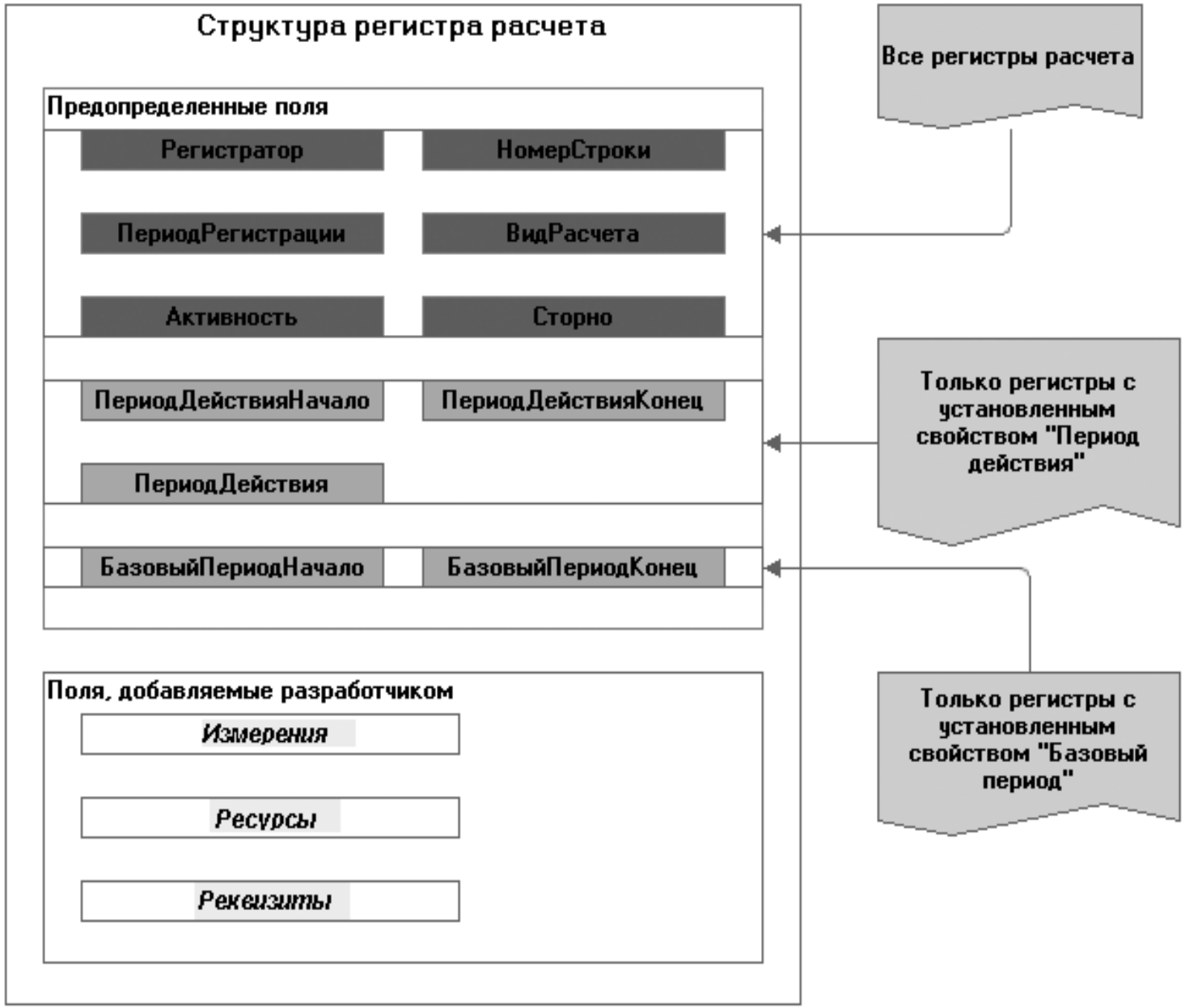
Свойства регистра расчета

- Свойство ***Период действия*** определяет, можно ли будет в данном регистре учитывать записи, протяженные во времени. Установленное свойство Период действия позволяет для каждой записи хранить не только период регистрации, но и период действия, отражающий протяженность этой записи.
- Кроме этого, установка этого свойства означает, что в данном регистре будет задействован механизм вытеснения и будет формироваться фактический период действия записей. Для того чтобы учитывать протяженные во времени записи, в плане видов расчета, используемом в данном регистре, должно быть установлено свойство ***Использует период действия.***
- Если в регистре будут учитываться записи, протяженные во времени, то необходимо настроить для данного регистра свойство ***График.***
- Свойство регистра расчета ***Базовый период*** позволяет хранить в регистре базовый период записей и, соответственно, использовать механизм зависимости по базовому периоду.

Неоднородность периода в графике



Структура регистра расчета



В простейшем регистре расчета без периода действия и базового периода всегда будут присутствовать следующие поля:

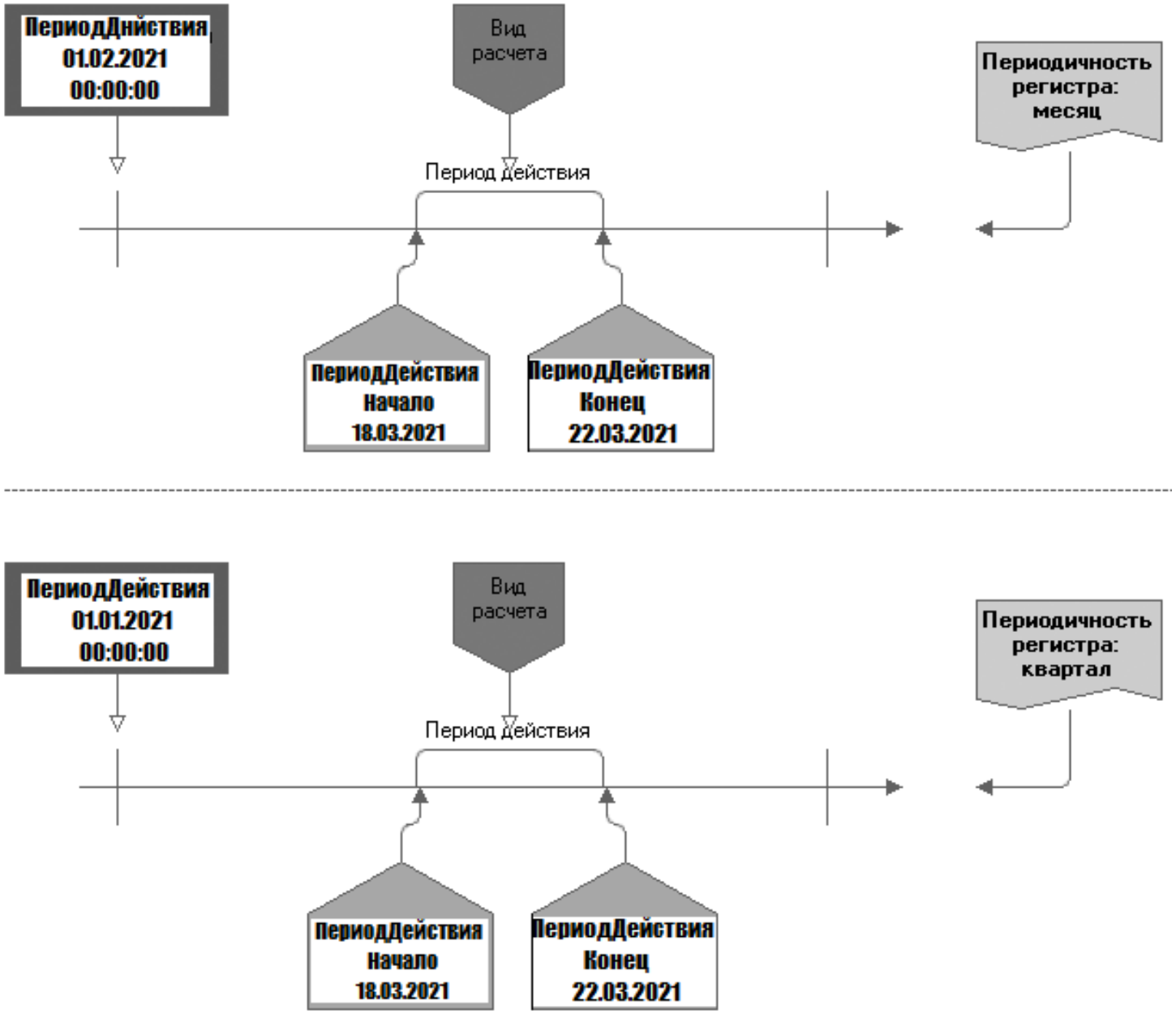
- ***Период регистрации*** – период регистра расчета, к которому относится данная запись. Размер периода определяется периодичностью регистра расчета.
- ***Вид расчета*** – ссылка на вид расчета, используемый в данной записи.
- ***Номер строки*** – порядковый номер записи в рамках данного регистратора. В рамках одного документа-регистратора номера строк уникальны. Таким образом, совокупность значений Регистратор и НомерСтроки позволяет идентифицировать конкретную запись регистра расчета.
- ***Активность*** – признак активности данной записи. Запись влияет на учет только в том случае, если активность установлена в значение Истина. Данное поле используется в документах прямой записи в регистр. В таких документах не хранятся данные, необходимые для формирования движений по регистру. Ввод записей производится напрямую в регистр, через вывод на форму документа таблицы с привязкой к набору записей этого регистра.

- ***Сторно*** – это признак типа Булево, который обозначает, является ли данная запись сторно-записью. Сторно-записи записываются в регистр со значением Истина в этом поле.

Если у регистра установлено свойство Период действия, то в структуре появляются следующие предопределенные поля:

- ***ПериодДействияНачало*** – дата, обозначающая начало интервала периода действия записи.
- ***ПериодДействияКонец*** – дата, обозначающая конец интервала периода действия записи.
- ***ПериодДействия*** – дата, отражающая период регистра расчета, в котором действовала запись. Эта дата всегда имеет значение начала первого дня соответствующего периода (по аналогии с полем ПериодРегистрации).

Пример заполнения полей «ПериодДействияНачало», «ПериодДействияКонец» и «ПериодДействия»



У регистров, для которых установлено свойство Базовый период, в структуре будут присутствовать следующие поля:

- **БазовыйПериодНачало** – дата начала интервала базового периода;
- **БазовыйПериодКонец** – дата окончания интервала базового периода.

Поля, добавляемые при разработке. Для каждого регистра расчета необходимо продумать структуру измерений, ресурсов и реквизитов, исходя из особенностей учетной задачи. Ввод ***измерений регистров*** расчета позволяет обозначить объект учета и расчета. При отсутствии измерений регистра расчетные механизмы действуют в рамках всех записей. Например, в этом случае любая запись о больничном приведет к вытеснению всех записей об окладе, для которых произойдет пересечение периодов действия. В результате добавления в регистр измерения ФизЛицо работа регистра изменится следующим образом: при вводе больничного по сотруднику Иванов будут вытеснены только записи об окладе по сотруднику Иванов за соответствующий период действия.

- Состав ***ресурсов регистра расчета*** влияет на возможность получения расчетной базы по данному регистру для записей данного регистра или других регистров. Например, если в регистре основных начислений есть ресурс Результат, хранящий рассчитанную сумму начисления, то для записей этого или другого регистра можно получить расчетную базу по этому полю, то есть сумму начислений всех записей по базовым видам расчета за базовый период. Если числовой показатель необходимо хранить в регистре, но в расчете по базе он не участвует, то такой показатель должен быть введен в структуру регистра как реквизит. Например, поле Размер начисления, отражающее тариф, по которому был рассчитан оклад или размер процента для начисления премии, не должно вводиться в регистр как ресурс, так как получать базу по этому полю бессмысленно. Ресурсы регистров расчета всегда имеют тип Число.
- ***Реквизиты регистра расчета*** – это произвольные дополнительные поля, которые необходимо хранить в регистре. Эти данные могут использоваться как при расчете записи, так и для формирования отчетов по регистру расчета. Например, реквизит РазмерНачисления участвует в формуле расчета записей, а реквизит СтатьяЗатрат может использоваться как группировка или отбор в отчете по начислениям. Добавление в регистр нового реквизита не оказывает принципиального влияния на учет и работу данного регистра (за исключением случаев, когда реквизит связан с графиком

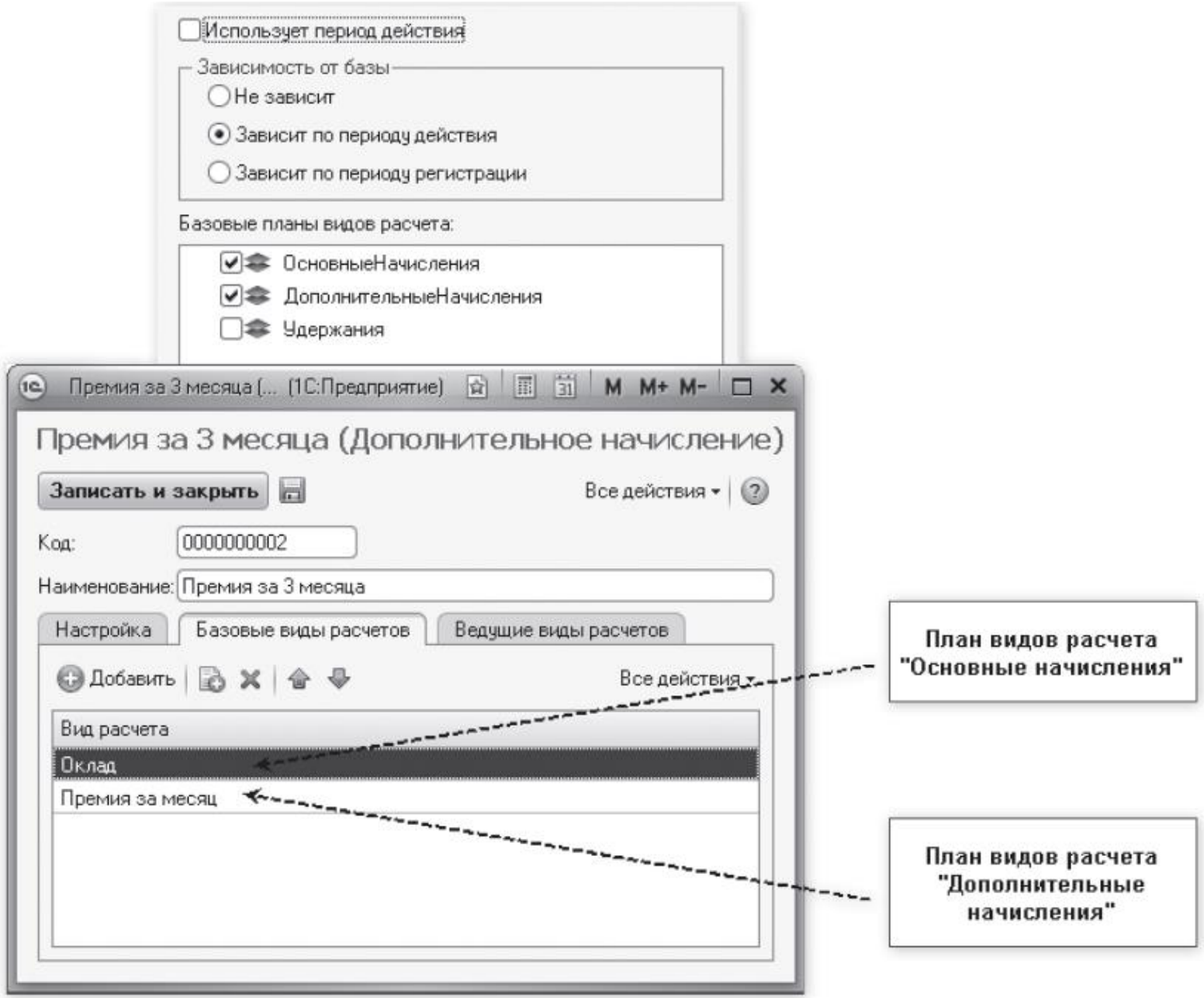
Настройка протяженных по времени расчетов: использование механизма вытеснения, использование графиков

- Для организации учета протяженных во времени расчетов у плана видов расчета должно быть установлено свойство ***Использует период действия***. Это позволит настраивать в видах расчета взаимное вытеснение путем заполнения предопределенной табличной части ***ВытесняющиеВидыРасчета.***
- В используемом регистре расчета должно быть установлено свойство ***ПериодДействия***, а также привязан график. При такой настройке все записи регистра помимо периода регистрации будут хранить данные о периоде действия, то есть о протяженности во времени.
- При формировании записи в регистре расчета, поддерживающем период действия, происходит анализ конкуренции за период действия вводимого вида расчета с существующими в регистре записями. Результатом такой конкуренции будет изменение фактического периода действия вводимой или существующих записей.
- Понятие ***фактического периода действия*** применимо только к конкретной записи регистра расчета. При этом данные о фактическом периоде действия не хранятся в основной таблице регистра расчета, но могут быть получены системными средствами.
- При формировании фактического периода действия происходит анализ конкуренции записей только в рамках данного регистра расчета. Иными словами, запись регистра расчета может вытеснять только записи этого же регистра.
- Протяженные во времени расчеты тесно связаны с данными графика, используемого в регистре расчета. Как уже отмечалось, данные графика содержат определенные значения для каждого календарного дня периода. Данные хранятся в системе в виде регистра сведений, созданного при разработке конфигурации. При этом дату графика платформа получает из измерения, заданного в свойствах регистра расчета в поле Дата графика, а значение графика – из ресурса, указанного в поле Значение графика.

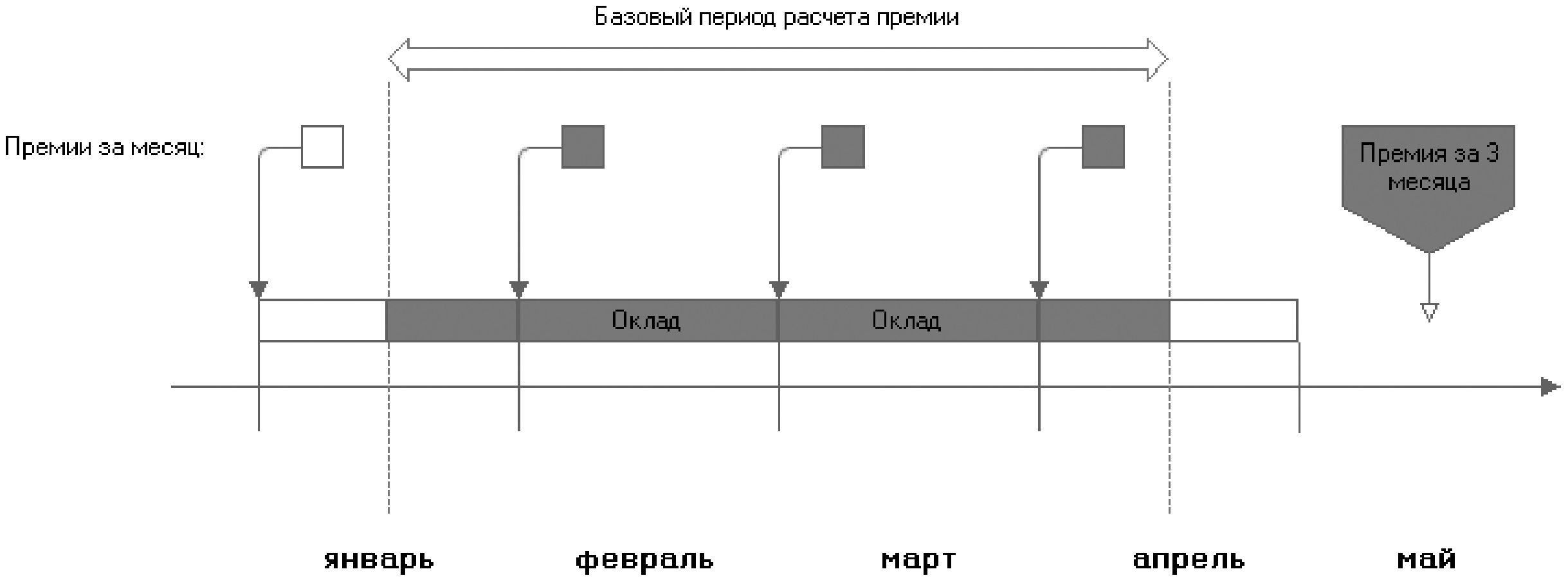
Настройка зависимости по базовому периоду

- В плане видов расчета, виды расчета которого будут зависеть по базовому периоду от других видов расчета, необходимо настроить зависимость от базы. Для этого свойство ***Зависимость от базы*** плана видов расчета должно быть установлено в одно из значений ***Зависит по периоду действия*** или ***Зависит по периоду регистрации.***
- ***Зависимость по периоду действия*** предполагает анализ пересечения фактического периода действия записей базовых видов расчета с базовым периодом текущей записи. При такой зависимости возможно частичное попадание какой-либо записи в базу расчета текущей записи. Если у записи по базовому виду расчета нет периода действия, то будет проанализировано попадание периода регистрации этой записи в базовый период текущей записи.
- В случае ***зависимости по периоду регистрации*** у базовой записи берется ее период, отсчитываемый от значения поля ПериодРегистрации. Например, при периодичности базового регистра Месяц будет анализироваться пересечение периода от даты ПериодРегистрации плюс месяц записей базовых видов расчета с базовым периодом текущей записи. Период действия базовых записей в данном случае значения не имеет.

Настройка зависимости по базовому периоду



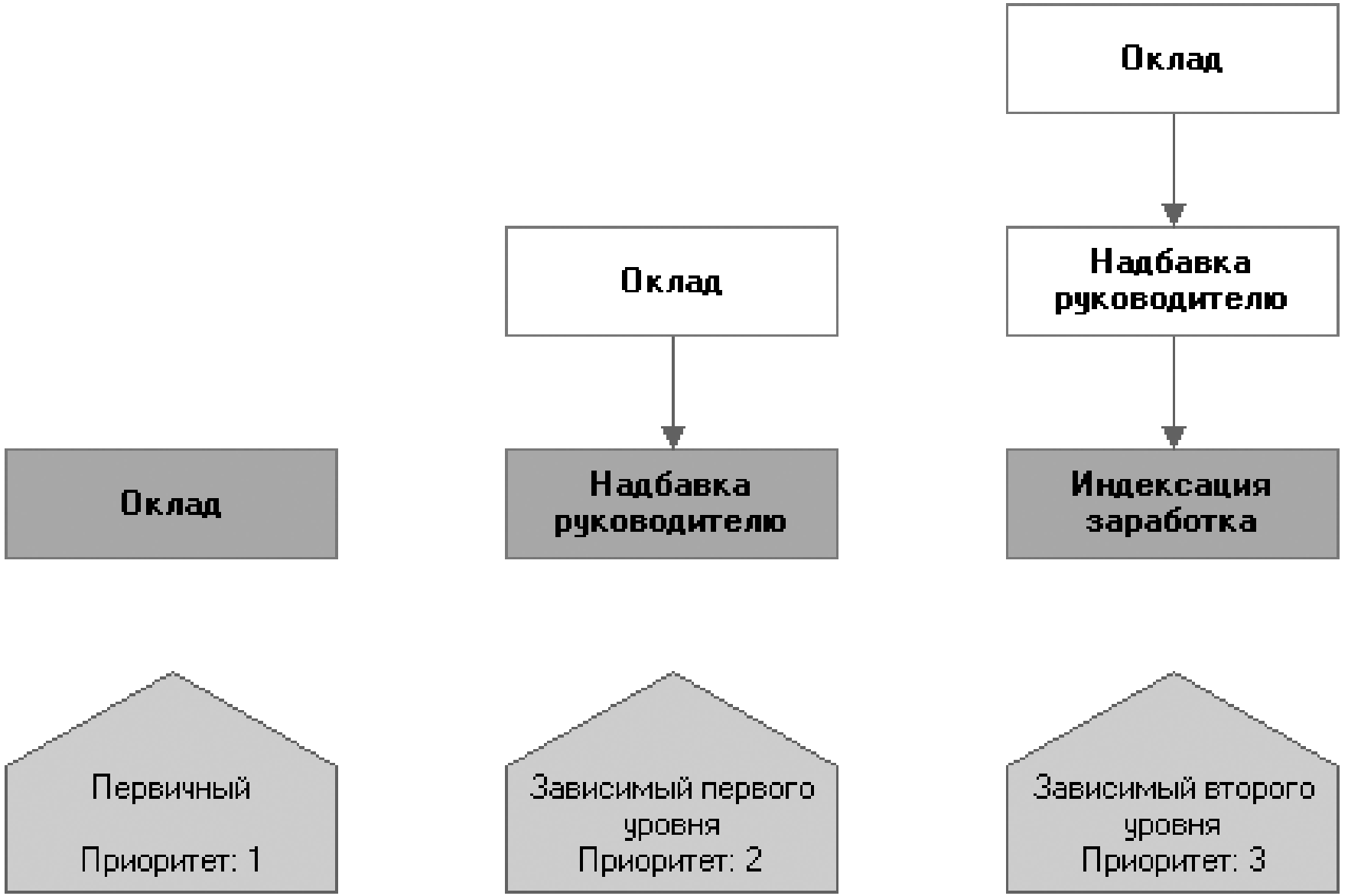
Зависимость от базы по периоду действия



- После настройки зависимости от базы необходимо указать, от каких видов расчета могут зависеть по базовому периоду виды расчета текущего плана видов расчета. Для этого нужно указать список базовых планов видов расчета в свойствах текущего плана видов расчета. В результате этой настройки в любом виде расчета текущего плана видов расчета в табличной части ***БазовыеВидыРасчета*** можно будет выбрать любой вид расчета отмеченных планов видов расчета.

Приоритет расчета записей

Приоритет видов расчета



Технология формирования и расчета записей регистров расчета

Схема расчета