**BC/NW 2017 № 2 (31):10.1**

**АНАЛИЗ АРХИТЕКТУР HLA, DIS И TENA ДЛЯ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ.**

Морозова Е.А., Мачарадзе Г.Т., Чернов С.А

**Введение**

В последние годы в связи с быстрым развитием информационных технологий широко используются компьютерные симуляторы, тренажеры и компьютерные моделирующие системы. Изначально данные технологии применялись только в военной сфере. Но на сегодняшний день существует много компьютерных игр, развивающих и обучающих тренажеров, которые используют данные технологии.

Толчок к развитию распределенных имитационных систем дало развитие аппаратных средств и развитие сети Интернет. Постепенно последовательные имитационные технологии начали заменяться на параллельные и распределенные.

Возникновение параллельных систем обусловлено появлением суперкомпьютеров с многопроцессорной архитектурой. К ним относятся симметричные мультипроцессорные системы SMP, массово-параллельные системы MPP, системы с неоднородным доступом к памяти NUMA и параллельно-векторные системы PVP. Для удешевления технологии MPP используются кластеры. Узлами в таких системах могут быть сервера, рабочие станции и персональный компьютер. Узлы связаны между собой по одной из стандартных сетевых технологий. Процесс кластеризации может осуществляться на любом уровне: аппаратном, системном и прикладном.

Распределенные системы же строятся из автономных независимых компьютеров. Главной отличительной особенностью распределенных систем является скрытие архитектуры системы, т.е. пользователь не видит сложных аппаратных средств, которые являются её основой. В качестве узлов распределенной системы могут выступать компьютеры, процессы или процессоры. На сегодняшний день под распределенной системой понимают локальные и глобальные сети, мультипроцессорные вычислительные системы и взаимодействующие процессы, функционирующие на одном компьютере.

На сегодняшний день существует много технологий построения распределенных имитационных систем. Но наиболее распространенными архитектурами являются HLA, DIS, TENA и другие.

## Принцип построения распределенных имитационных систем

Основными преимуществами систем имитационного моделирования является предоставление пользователю естественной средой для разработки имитационных моделей и предоставление дополнительных возможностей.

При выборе распределенной системы моделирования в числе основных требований, предъявляемых к среде разработки, является отказоустойчивость, синхронизация времени и организация взаимодействия объектов модели в процессе моделирования.

В настоящий момент существуют разные модели, методы и средства распределенного имитационного моделирования.

Новым подходом к реализации распределенных систем имитационного моделирования является стандарт HLA. Данный стандарт был разработан оборонным отделом имитации и моделирования (DMSO) при Министерстве обороны США. Стандарт построен на инфраструктуре времени выполнения RTI, которая включает серверную и клиентскую части, и библиотеки классов JavaBinding, предназначенная для обеспечения взаимодействия между распределенными моделями реализованных на Java. Основными преимуществами HLA является поддержка различных механизмов синхронизации времени, открытая спецификация, возможность применения, как в параллельном, так и в распределенном моделировании.

В общем случае схему выполнения процесса в распределенной имитационной модели можно представить следующим образом (рис. 1).

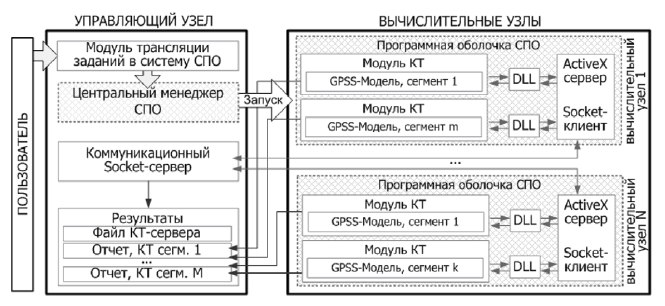


Рисунок 1 - схема процесса выполнения распределенной имитационной модели

В данной схеме исполнительная среда имеет двухуровневую архитектуру управления вычислительным кластером. Управляющая программная настройка, функционирующая на верхнем уровне, транслирует запросы от генератора распределенной модели в формат системы пакетной обработки. Управление запуском задания осуществляется на нижнем уровне модулями системы пакетной обработки.

## Архитектура DIS

Распределённый интерактивный симулятор (DIS) – это стандарт протокола, который даёт возможность участников компьютерных симуляций взаимодействовать с виртуальным миром. Он предназначен для совместного использования возможностей виртуальной среды. Данная система обеспечивает взаимодействие участников только внутри модели. Внешние взаимодействия не входят в данную архитектуру, так как они слишком многообразны и велики.

При построении виртуальной среды в архитектуре DIS необходимо определить, какие объекты будут неактивными или малоактивными. В виртуальной среде определить это проблематично. Поэтому, разработчикам приходится решать, какие объекты отображать, а какие нет. Это необходимо для экономии мощности компьютеров. Таким образом, в виртуальных мирах существует главная проблема – определение набора элементов существующих в системе и ранжирование их по значимости.

Для архитектуры DIS для решения данной проблемы разработана технология фильтров, которые отсеивают ненужные объекты и их действия. Этими фильтрами отсеиваются в основном мелкие объекты, чьи действия и передвижения незаметны на больших дистанциях.

Технологии фильтров основаны на таблицах, которые описывают саму виртуальную среду и элементы. Описание среды стоится по принципу вложенных таблиц. Например, имеется глобальная таблица, описывающая мир в целом. В данной таблице каждая ячейка, описывающая определенную область, содержит другую таблицу, которая содержит более подробное описание данной области и т. д. Количество вложенных таблиц теоретически может быть бесконечным, но практически имеются ограничения.

По сути DIS похожа на бегущую строку комментариев в футбольном матче, которая представляет собой обычную передачу по сети знаков. Принцип работы DIS аналогичен деятельности комментатора, который наблюдая за игрой должен отображать происходящее несколькими короткими фразами. Все игроки имеют номера и имена, их позиция передается с помощью коротких фраз. Сообщения, которые доходят до слушателя, невидящего игру, и создают у него представление происходящего. Таким образом, слушатель ярко представляет картину происходящего, слушая монолог комментатора.

Эта же идея лежит в основе DIS. Компьютер, получая сообщения по заранее определенным правилам, может отобразить модель происходящего на экране. Каждый момент симуляции DIS упаковывает в сжатую форму и передает по сети. Разница между симуляцией в DIS и футбольным матчем заключается лишь в разнообразии моделей и размерами игрового поля.

В архитектуре DIS игроками являются определенные сущности, имеющие независимое представление. В качестве сущности могут выступать танк, джип, самолёт, человек и т.д.

Каждая сущность определена коротким кодом с указанием модели, серийный номер и другие отличительные характеристики. Перемещение сущностей описывается вектором, содержащего показатели скорости, координат и других физических величин (рис. 2). Такие действия, как радиопередача, стрельба, ремонт, перезагрузка и другие имеют свой собственный код. Таким образом, каждое имитационное действие кодируется как специально форматированный пакет данных, который и является протоколом PDU.

Последовательность событий в симуляторе кодируется как поток PDU-кодов, которые доставляются всем компьютерам принимающих участие в данной симуляции. При получении PDU-сообщения, каждый симулятор свободен в интерпретации полученных данных. Так, например, обычная модель машины покажет реальную картину движения.

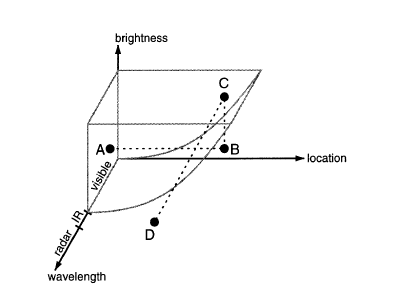


Рисунок 2 – Принцип взаимодействия сущностей:

подписи осей: wavelength –расстояние,brightness – яркость, location – текущее местоположение

В отличие от футбольного матча, где отправитель сообщений один комментатор, в системе DIS каждая сущность создает свои сообщения. Кроме того каждая сущность может играть роль как комментатора, так и слушателя одновременно. Все участники симуляции посылают свои сообщения в общий канал и слушают его. На этот канал может настроиться любой желающий и получить все индивидуальные сообщения.

Существуют два режима передачи сообщений в DIS:

* общий – сообщение передаётся всем компьютерам;
* индивидуальный – сообщение передаётся конкретному компьютеру.

По широковещательному каналу компьютеры рассылают данные каждому компьютеру в сети. Данная модель хорошо работала в ранних версиях DIS и хорошо себя зарекомендовала.

Но со временем пользователи стали предъявлять более высокие требования к симуляциям. Симуляции стали использовать большие игровые поля, распределенные на тысячи километров, появилось множество сущностей, дополнительные эффекты и т.д. Все это вызвало перегрузку имеющихся каналов связи. Для работоспособности DIS в таких условиях, поток данных необходимо сдерживать.

**Архитектура HLA**

Идеология HLA объединяет множество объектов, которые принимают участие в распределенном моделировании, в федерации – динамически формируемая сущность. Объекты, входящие в состав федерации, получили название федератов. Федерации и федераты являются логическими понятиями. В качестве федератов могут быть тренажеры, реальная техника, люди, автоматизированные командные системы, легионы виртуальных войск и т.д. Системы формирования виртуального пространства являются особым классом федератов.

Взаимодействие федератов в архитектуре HLA осуществляется инфраструктурой реального времени RTI (рис. 3). В состав RTI входят сервисы, которые поддерживают в едином модельном времени местоположение федератов и взаимный обмен данными.

Рассмотрим пример, в котором в качестве федерата выступает имитационная модель истребителя (рис. 4). Механизм RTI позволяет передавать значения высоты, скорости, траектории полета для остальных участников федерации. Если возникает необходимость, то может передаваться и аудио и визуальный образ федерата, а также его тактико-технические параметры. Таким образом, командир имеет возможность наблюдать перемещения данного самолета на общей карте боевых действий, солдат в танковом тренажере видит проносящийся над ним самолет, а диспетчер аэродрома в симуляции проводит переговоры с пилотом.

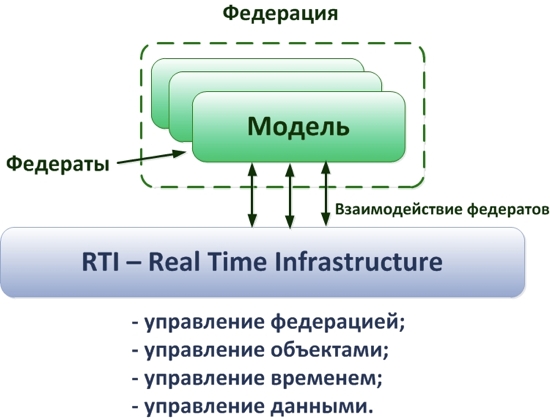


Рис. 3 – Инфраструктура RTI



Рис. 4 – Пример симуляции с использованием механизма RTI

Детализация виртуальной действительности зависит от возможностей технических средств и полноты федерации. Иногда бывает достаточно показать координаты сил, которые ведут бой, а порой необходимо изменить ландшафт вследствие попадания снаряда в какой либо объект.

RTI так же предоставляет федератам и другие сервисы. Например, подключение и выход из состава федерации, изменения атрибутов, передача прав владения атрибутами и т.д. Все сервисы, входящие в состав RTI имеют строго прописанный интерфейс. В результате этого в рамках платформы HLA обеспечивается интероперабельность федератов и RTI, т. е. особенности языка и платформы не существенны.

Так как HLA является высокоуровневой архитектурой, то, как и все протоколы высокого уровня, она не требует каких либо ограничений от федератов и RTI. HLA представляет собой набор рекомендаций по форматам данных, которые могут использовать участники симуляции, и правила взаимодействия федератов при различных условиях. Соблюдая эти предписания и правила, разработчики могут реализовывать модели, используемые в дальнейшем в различных моделирующих комплексах, и собственные варианты инфраструктуры RTI. На сегодняшний день существуют готовые решения RTI как коммерческих, так и свободно распространяемых. Наибольшую популярность завоевали MAK-RTI (американская версия) и Pitch-RTI (европейская версия). ОАО «НПО РусБИТех» реализовал RTI для России, которая не уступает по производительности западным аналогам при более низкой стоимости. Отечественная версия RTI апробирована при решении различных задач интеграции разрабатываемых и уже созданных тренажеров и моделирующих систем на основе HLA-технологии.

В архитектуре HLA имеется стандартная модель описания объектов OMT. Данная модель содержит описания объектов, федератов и федераций. Модель OMT служит для:

* предоставления механизма определения обмена данными и общей координации среди членов федерации;
* обеспечения общего, стандартного механизма описания возможностей потенциальных участников федерации;
* упрощения разработки и применения общих инструментальных средств для HLA моделей.

Структура архитектуры описывается тремя компонентами:

* моделью OMT;
* правилами федерации;
* спецификациями интерфейсов.

Модель OMT позволяет задокументировать информацию о моделях объектов и тем самым обеспечивает выполнение требования интероперабельности и многоразового использования федератов и их компонентов.

Все внешние объекты и их взаимодействие, доступные другим федератам, должно быть подробно описано в формате OMT. Для этого в архитектуре HLA используются базовые понятия класс, иерархия классов, объект, атрибут, интеракция и параметр.

Совокупность объектов с общими характеристиками образуют классы. Экземпляр это каждый индивидуальный объект в данном классе. Классы позволяют упростить процесс создания необходимых организационных схем и определяются разработчиками моделей. Каждый класс имеет уникальный набор характеристик, с помощью которых задается экземпляр класса.

При наличии набора характеристик одного класса, которые являются подмножеством характеристик другого класса, образуют расширение, либо наследуют характеристики, либо являются потомками первого класса. Такое отношение называют иерархией классов.

В архитектуре HLA используются два типа сущностей: объект и интеракции, поэтому присутствуют классы объектов и интеракций.

Объект в архитектуре HLA определяется по аналогии с объектами объектно-ориентированного программирования – модель любого явления из реального мира. Отличительной чертой объектов HLA является отсутствие методов. У объектов имеются только состояния. Практически объект HLA представляет собой только структуру данных без функций обработки.

Каждый класс объектов определяется стандартным набором атрибутов, которые определяют состояние объектов. Их значения могут меняться со временем.

Под интеракцией понимают действия, воспроизводимых с одним или набором объектов, которые отражаются на другом или других объектах. Любой класс интеракций определен уникальными параметрами, значения которых определяют разные экземпляры интеракций одного класса. У интеракций нет состояний, поэтому и нет атрибутов. По своей сути интеракция это информационное взаимодействие.

Все выше перечисленные понятия применяются в основных моделях, которые используются в HLA: объектной модели федерации FOM и объектной модели имитации SOM.

Каждая федерация в архитектуре должна иметь свою модель FOM. На основании данной модели классифицируются все объекты и интеракции, управляемые данной федерацией, и хранятся детальное описание всей информации по классам и их характеристик. Этой информацией могут обмениваться федераты. Модель так же нужна для идентификации интерфейсов между федератами, RTI и сервисами RTI.

Любой объект и интеракция из федерации принадлежат к конкретному классу объектов или классу интеракций из модели FOM. В то же время модель FOM включает только те классы и характеристики, которые в ходе использования федерации обмениваются участники.

В рамках федерации, федералы могут устанавливать любому объекту или интеракции дополнительную информацию, не доступную другим федералам. Такая информация не будет отражена в объектной модели федерации. Кроме того, для своих нужд федералы могут создавать собственные объекты и интеракции, которые не будут принимать участие в информационном обмене между федералами. Такие классы так же не будут присутствовать в объектной модели FOM.

Объектная модель федерации предоставляет описание правил обмена данными между участниками в стандартизированном виде на уровне классов. Это необходимо для обеспечения взаимодействия федералов.

Для характеристики каждого федерала используется объектная модель имитации SOM, которая так же позволяет характеризовать все выполняемые им действия в рамках федерации. Данная модель содержит группу классов объектов, интеракций и их характеристики из объектной модели федерации, к которой относится данный федерат. Федерат может либо сам управлять объектами, либо получать атрибуты и параметры объектов от других федератов. Таким образом, объектная модель имитации содержит только описание внешних возможностей федерата. В ней отсутствуют описание внутренних возможностей действующие в рамках федерации.

Объектная модель имитации позволяют на этапе проектирования существенно упростить анализ применения конкретных имитационных систем для использования в федерации. Она является базой для создания объектной модели федерации.

Таким образом, модель SOM позволяет описать каждого федерала, который входит в модель, а модель FOM – модель самой федерации.

Модель OMT содержит стандартные модели FOM и SOM, которые не зависят от их конкретного информационного наполнения. Вся информация в модели OMT должна быть представлена в виде таблиц:

* структуры классов объектов, содержащие имена всех классов объектов федерата или федерации и их иерархии «класс–подкласс»;
* структуры классов интеракций для хранения имен всех классов интеракций федерата или федерации и для описания их иерархии «класс–подкласс»;
* атрибутов, определяющих атрибуты классов объектов федерата или федерации;
* параметров, определяющих параметры классов интеракций федерата или федерации.

Кроме этой информации в модели могут содержаться расширения модели, например, уровень точности, с которой классы и все их характеристики представляют реальный мир; метаданные объектных моделей SOM/FOM и др.

Благодаря независимости HLA от конкретной реализации, стало возможным проводить сложные виртуальные учения с участниками стран различных коалиций и многократно использовать модели, которые являются виртуальным аналогом дорогостоящего оборудования с помощью аренды в рамках федерации.

Несмотря на то, что HLA построена на базе протокола DIS, она не совместима со своим предшественником. Однако модели, построенные на основе данной технологии, могут взаимодействовать между собой. Разработано множество программных модулей, которые обеспечивают попадание снаряда, выпущенного на полигоне под управлением протокола DIS, в цель на поле боя в архитектуре HLA.

Несмотря на достоинства данной архитектуры так же имеются и недостатки:

* сложность стандарта;
* определенный предел масштабируемости;
* нужны дополнительные средства для обеспечения отказоустойчивости;
* модели из различных версий RTI не могут взаимодействовать между собой.

## Стандарт (архитектура) TENA

Стандарт TENA был опубликован в 2001 году и его основное назначение обеспечения взаимодействие Министерства обороны США с тестовыми и обучающими системами. Данная архитектура предназначена для проведения комплексных тестирований и моделирований при использовании крупномасштабных распределенных в реальном времени виртуальных средах, интегрирующих тестирование, обучение, моделирование с использованием единой архитектуры (рис. 5).

Главная цель стандарта – обеспечение архитектуры и реализации программного обеспечения необходимого для:

* обеспечения эффективного взаимодействия между комплексами, объектами моделирования и систем C4ISR;
* многократного использования моделей;
* обеспечения модульности, инициализации и тестирования.

TENA включает в свой состав пять основных приложений.

TENA Applications. В него входят наборы инструментов, ресурсов, системы обработки данных, которые построены с применением архитектуры TENA. Инструменты, содержащиеся в стандарте, многоразовые, хранятся в базе данных и доступны для всех участников симуляции.

Non-TENA Applications. Содержит наборы систем обработки, тестовых систем, симуляций и c4isr системы, которые не входят в стандарт, но нужны для обеспечения тестирования и обучения.

TENA CommonInfrastructure. Это подсистемы программного обеспечения, которые обеспечивают функционирование данной архитектуры. К ним относятся:

* TENA Repository для хранения приложений, объектных моделей и другой информации;
* TENA Middlewareдля обмена информацией в режиме реального времени;
* EventDataManagementSystem, которая обеспечивает хранение сценариев, сведений, данных, полученных в результате симуляции, и краткую информацию.

TENA ObjectModels – общий стандарт для обеспечения взаимодействия между всеми ресурсами и инструментами. Набор объектов, участвующих в симуляции называется областью объектной модели (LROM) и содержит стандартные и пользовательские определения объектов.

TENA Utilities – содержит утилиты, которые обеспечивают визуализацию и управление симуляциями архитектуры.

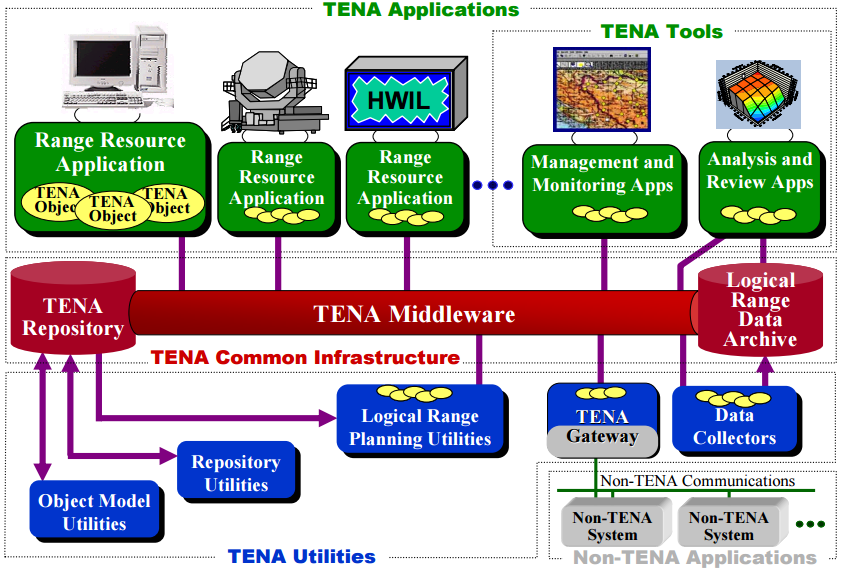


Рис 5 – Модель взаимодействия стандарта Tena

**Заключение**

Имитационное моделирование с развитием информационных технологий так же развивается. В последние десятилетия центр исследований от последовательных систем был смещен в сторону распределенных, а в последние годы – в сторону виртуального моделирования.

Бурное развитие имитационного моделирование обусловлено, прежде всего, интересами военных в данной области, хотя область применения имитационных моделей очень разнообразна.

Архитектура DIS является прародительницей архитектуры HLA. Данная архитектура имеет один существенный недостаток - необходимость сдерживания потока данных, что реализуется фильтрами.

Архитектура HLA получила широкое распространение не только в военных симуляторах, но и в других сферах. Однако данная архитектура слишком сложна, имеет предел масштабируемости, нужны дополнительные средства для обеспечения отказоустойчивости и модели из различных версий RTI не могут взаимодействовать между собой.

Стандарт TENA предназначен для проведения комплексных тестирований и моделирований при использовании крупномасштабных распределенных в реальном времени виртуальных средах, интегрирующих тестирование, обучение, моделирование с использованием единой архитектуры. В основном данный стандарт используется в военных и игровых симуляторах.

Область применения распределенных систем моделирования не ограничивается только военными разработками.

Как HLA, так и TENA являются «потомками» DIS, однако стандарт HLA совместим со стандартом DIS и только частично со стандартом TENA (TENA имеет шлюзы и общий функционал для работы с другими стандартами, например, карта событий, у которой есть система координат, типы сущностей и многое другое, но не может быть полностью с ними совместим).

На выше рассмотренных архитектурах функционируют обучающие системы и тренажеры для подготовки специалистов различных специальностей.

Распределенные имитационные системы с успехом используются так же в психологии, социологии, экономике и в научной сфере.

**Литература**

1. Bakhmurov A. G., Balashov V. V., Pashkov V.N., Smeliansky R.L., Volkanov D. Yu. Method For Choosing An Effective Set Of Fault Tolerance Mechanisms For Real-Time Embedded Systems, Based On Simulation Modeling // Problems of dependability and modeling/eds. JacekMazurkiewicz [i in.]. Wrocіaw // OficynaWydawniczaPolitechnikiWrocіawskiej. 2011. – P. 13–26.
2. Dalsgaard A., Olesen M., Toft M., Hansen R., Larsen,K. METAMOC: Modular Execution Time Analysis using Model Checking // SchlossDagstuhl – Leibniz-ZentrumfuerInformatik, Germany. 2010 – P. 113–123.

High Level Architecture – https://www.dmso.mil/public/transition/hla

High Level Architecture – https://www.dmso.mil/public/transition/hla/

1. J. Russell Noseworthy, Ph.D. LeadThe Test and Training Enabling Architecture (TENA)—Supporting the Decentralized Development of Distributed Applications and LVC Simulations//TENA SDA Software Development Lead. 2013

The Object Management Group – http://www.omg.org/

Александров А. А. Распределённое имитационное моделирование: технологии, методы, средства// Вестник НГУ, 2011, т.7 выпуск 3 –8 с.

1. Бигдан В.Б., Пепеляев В.А., Сахнюк М.А. О методолого-технологических аспектах последовательного имитационного моделирования//Материалы VI всероссийской научно-практической конференции ИММОД–2011, ФГУП ЦНИИ технологии судостроения, Санкт-Петербург . c.170-176
2. Бродский Ю. И. Распределённое имитационное моделирование сложных систем: Монография// Вычислительный центр им. А.А. Дородницына Российской академии наук Москва. 2010 – 32 с.