ООО «Алгоритм»

Разработчик отечественного ПО

КОМПЛЕКС ПРОГРАММНЫХ СРЕДСТВ «BUSINESS SCENARIOS CINEMA»

Описание технической архитектуры ПО

Москва
2024

Оглавление

Общие положения

1. Настоящая инструкция выпущена с целью обеспечить пользователя необходимой информацией о технической архитектуре программы Cinema.
2. Описывает ключевые алгоритмы работы ядра и схему взаимодействия компонентов ПО.
3. Рекомендуется для ознакомительного чтения перед выполнением любых действий в программе.

Термины и определения

ПО – программное обеспечение.

Cinema – комплекс программных средств для быстрой разработки сервисов, их управления и мониторинга. Сервис представляет из себя сценарий обработки данных в специализированном формате, описывающий участников (плагины), их конфигурацию и взаимосвязи между ними.

Компоненты комплекса Cinema:

Cinema – ядро – основной блок ПО, поставляемый в виде библиотеки (Cinema.dll/libCinema.so), предназначенный для загрузки и исполнения сценария сервиса.

Actor – приложение, предназначенное для выполнения различных сценариев обработки данных через подключённое ядро Cinema.

CinemaNode – приложение управления запуском и остановки множества сценариев на одном сервере, контроля их работоспособности через приложение Actor.

Hypervisor – приложение, предназначенное для создания сценариев обработки данных, их удалённого выполнения и мониторинга.

Библиотека плагинов – структурированный набор плагинов различного назначения, на основе которых строятся сценарии сервисов обработки данных.

Ключевые принципы алгоритма работы ядра и разработки плагинов

1. Для возможности предоставления плагина внешним потребителям (ядру Cinema) библиотека экспортирует два метода:
2. GetPluginList – список реализованных плагинов в библиотеке.

3. CreatePlugin2 –создание плагина по имени, указанному в параметрах метода.
4. Количество реализуемых плагинов в одной библиотеке не ограниченно.
5. В момент выполнения сценария, количество создаваемых экземпляров плагинов одного вида равно количеству применённых в сценарии.

Пример:

Сценарий, где в процессе выполнения, может быть создано максимум 11 плагинов Client. (См. рисунок 1).

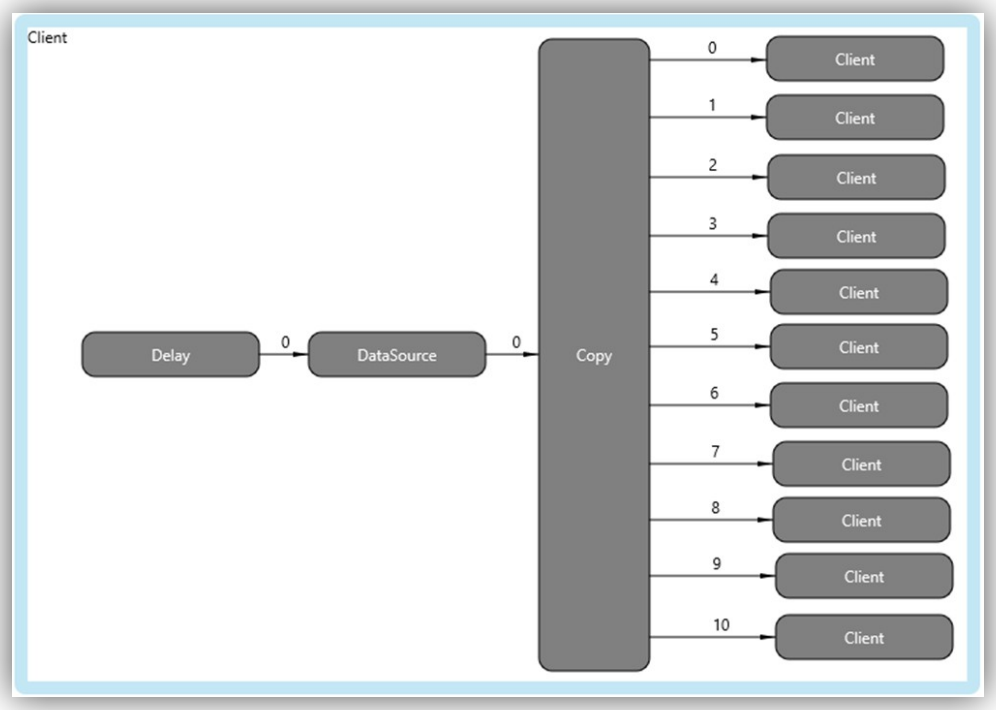

Рис. 1

Общая схема взаимодействия компонентов Cinema

1. Взаимодействия компонентов Cinema в частной сети – это комплекс, с котором Cinema является сервисом, описанным конкретным сценарием и выполняемым приложением Actor с помощью ядра Cinema.
2. Каждый сервер Node управляется через CinemaNode.
3. CinemaNode выполняет команды, получаемые от Hypervisor, а обратно передает статистику о выполняемых на текущий момент сервисах (Actor).

4. CinemaNode контролирует состояние и обеспечивает непрерывный режим работы запущенных сервисов. Запускает сервисы заново в случае внезапного закрытия или зависания.
5. Оператор, через приложение Hypervisor, отслеживает состояние всех сервисов на выбранном кластере серверов. Имеет возможность добавлять и запускать новые, останавливать и удалять существующие. (См. рисунок 2).

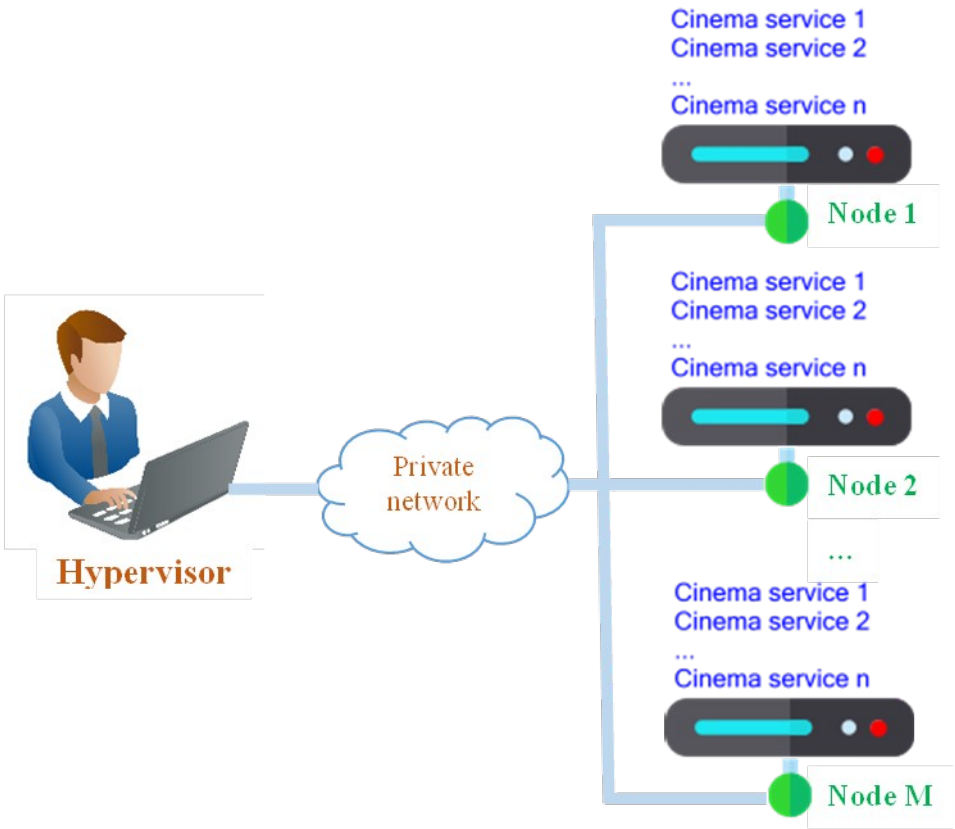

Рис. 2

6. Большинство компонентов Cinema реализованы на языке программирования Pascal и компилируются с помощью свободно распространяемого под лицензией GNU GPL2 компилятора Free Pascal Compiler. Разработка плагинов возможна так же на языке C++.
7. Возможности Cinema целиком и полностью зависят от набора плагинов в поставке дистрибутива. Каждый плагин выполняет определённую работу. Плагин в сценарии может быть связан с другим плагином связью, что означает что после выполнения своей работы, ядро передаст его результат следующему по связи плагину.

Пример:

Сценарий распознавания текста в изображениях.

- Плагин PathReader загружает изображения из заданной директории и передаёт их далее на выход, плагину OCR согласно сценарию.
- Плагин OCR выполняет распознавание текста и, выделенный из изображения текст, отправляет на свой выход, соединённый со входом плагина FileStreams.
- Плагин FileStreams выполняет сохранение поступивших на вход данных в заданную директорию. (См. рисуно 3).

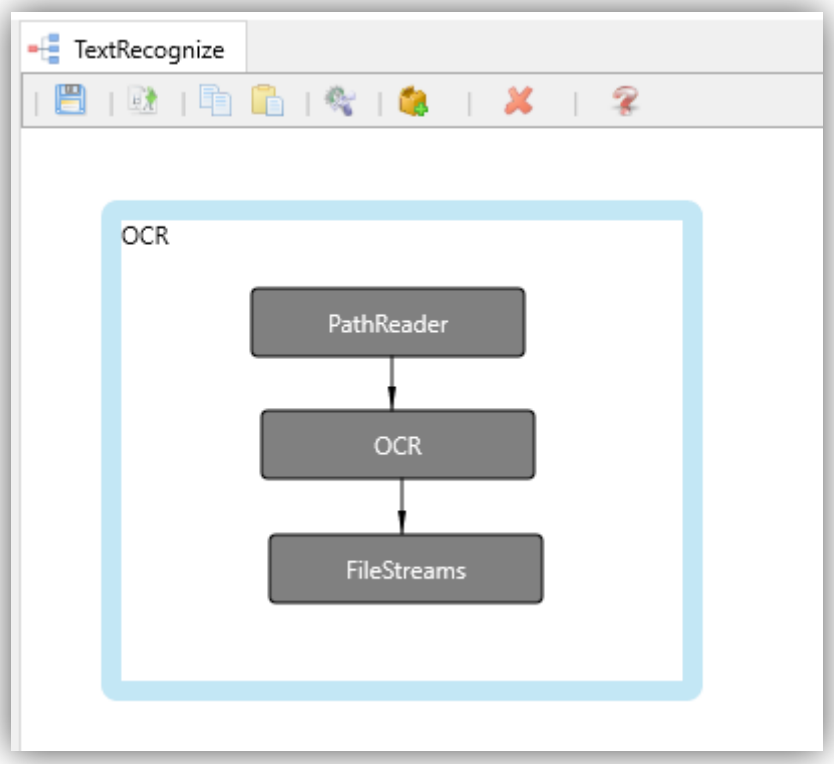

Рис. 3

8. Плагины поставляются из библиотек. Одна библиотека может содержать неограниченное число плагинов. Рекомендуется при создании и добавлении плагинов располагать их по принципу области применения.

Пример: библиотека Network содержит набор плагинов работы с сетью и протоколами передачи данных.

9. Архитектура ядра Cinema представленна на рисунке 4. Схема является условной, так как отображает алгоритм выполнения сценария.

    - Каждый плагин реализует интерфейс ICinemaPlugin, через которое ядро передаёт плагину входные данные. Результат работы плагин отправляет через экземпляр реализации интерфейса ICinema, получаемый им при его создании.

- Scenario Thread Layer, реализующий ICinema, является отдельной единицей выполнения сценария – системный поток. Их количество задаётся в момент создания сценария. На рисунке 4 представлен один поток, отображаемый в виде бирюзового прямоугольника и подписанный как OCR.
- Если потоков несколько, и существуют связи между плагинами из различных потоков, тогда ядро в момент загрузки для таких связей создаёт очереди типа FIFO – First In First Out – первым пришёл первым ушёл. В таком потоке данные сначала размещаются в очереди, а затем вычитываются из неё принимающим потоком, и после отдаются соответствующему плагину на обработку.
- В момент выполнения сценария плагины создаются при первом обращении к ним ядра. Первыми создаются «стартовые» плагины – это плагины, у которых по сценарию нет входной связи. На рисунке 4 это плагин PathReader. С PathReader начинается выполнение сценария.
- Каждый плагин может иметь неограниченное количество входных и выходных физических Разработчик описывает связи в спецификации плагина. На каждом физическом выходе плагин может создать неограниченное число виртуальных потоков данных (Data Stream), каждый из которых может параметризировать. Например, плагин Server. Он принимает данные от множества клиентов и для каждого клиента создаёт виртуальный поток данных, параметризируя его данными о клиенте: IP адрес клиента и исходящий порт. Последующие плагины так же принимают и обрабатывают уже разделённые потоки данных, при этом им доступно «описание» каждого потока в виде его параметров.
- Любой плагин может добавить свои параметры потока данных, например, ставшие ему известными в момент обработки этого потока. (См. рисунок 4).

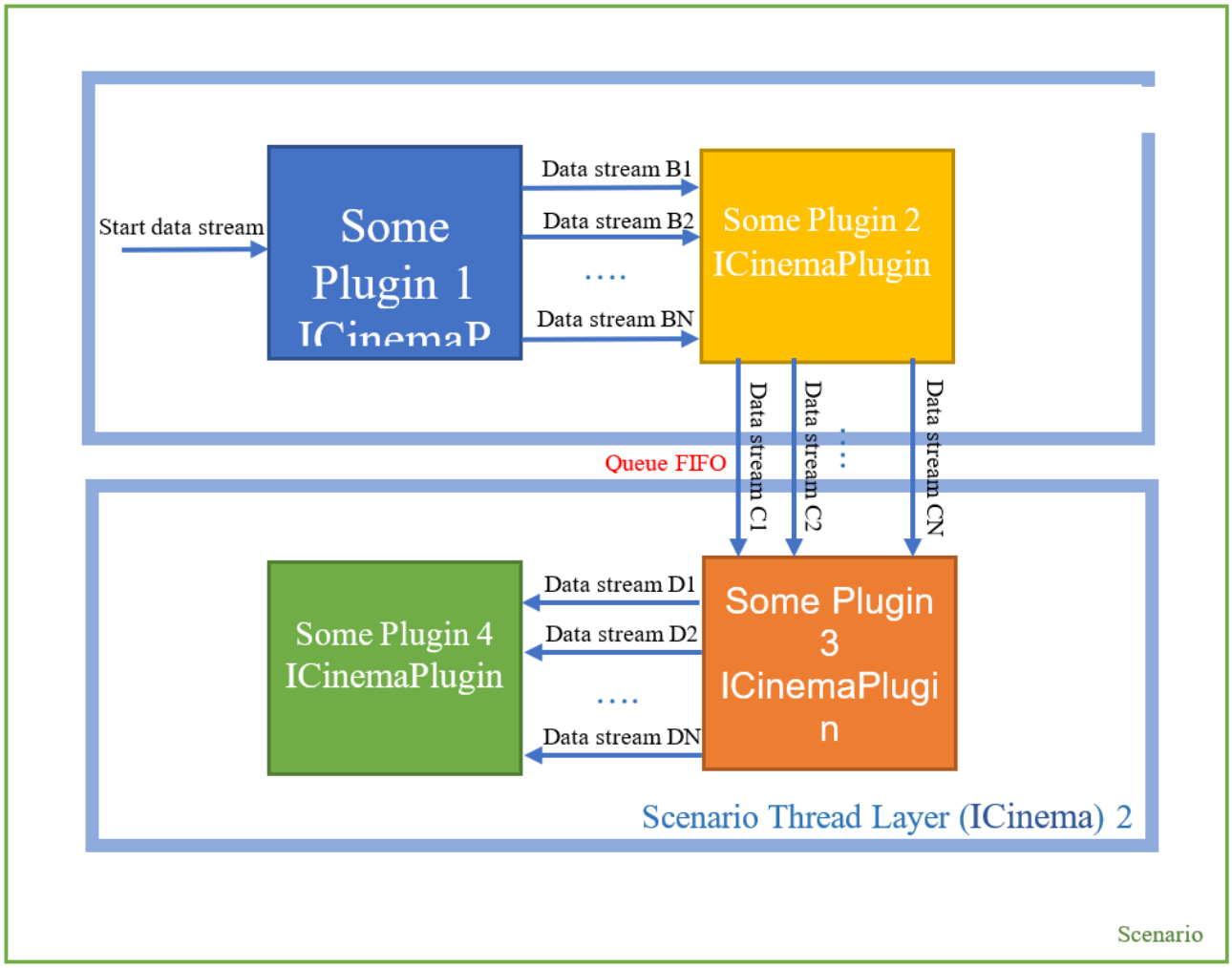
Start data stream
Some Plugin 1 ICinemaP
Data stream B1
Data stream B2
....
Data stream BN
Some Plugin 2 ICinemaPlugin
Data stream C1
Data stream C2
....
Data stream CN
Queue FIFO
Some Plugin 4 ICinemaPlugin
Data stream D1
Data stream D2
....
Data stream DN
Some Plugin 3 ICinemaPlugi n
Scenario Thread Layer (ICinema) 2
Scenario

Рис. 4