

МЕТОДИКА ПОДКЛЮЧЕНИЯ ПРОГРАММНЫХ МОДУЛЕЙ К СИСТЕМЕ ЛИЦЕНЗИРОВАНИЯ ПАКЕТА ПРОГРАММ ЛОГОС

*Губайдуллина Елена Алексеевна (eskova.ea@mail.ru), Дьяков Александр Владимирович,
Тишкин Дмитрий Александрович*

ФГУП «РФЯЦ-ВНИИЭФ», г. Саров Нижегородской обл.

В работе представлены результаты внедрения системы лицензирования в программные модули, входящие в состав программ ЛОГОС, в виде разработанной методики подключения программных модулей.

Ключевые слова: пакет программ ЛОГОС, система лицензирования, сервер лицензий.

A PROCEDURE OF CONNECTING PROGRAM MODULES TO THE LICENSING SYSTEM OF THE LOGOS SOFTWARE PACKAGE

*Gubaydulina Elena Alekseevna (eskova.ea@mail.ru), Dyakov Aleksandr Vladimirovich,
Tishkin Dmitriy Aleksandrovich*

FSUE «RFNC-VNIIEF», Sarov Nizhny Novgorod region

The report presents the results of introducing the licensing system to the program modules of the LOGOS software package in the form of the procedure of connecting program modules.

Keywords: the LOGOS software package, licensing system, license server.

Введение

Пакет программ ЛОГОС [1] – это коммерческий цифровой продукт ФГУП «РФЯЦ-ВНИИЭФ», для инженерного анализа и суперкомпьютерного моделирования, позволяющий решать задачи математического моделирования в разных сферах науки и техники. Пакет программ ЛОГОС состоит из различных программных модулей и отдельных библиотек, использование которых требует соблюдения определенных лицензионных соглашений.

Система лицензирования ЛОГОС является универсальным средством контроля за соблюдением условий лицензионного соглашения, поэтому возможно ее внедрение в другие разрабатываемые программные продукты ФГУП «РФЯЦ-ВНИИЭФ» в рамках ЛОГОС.

В данном докладе описана методика подключения программных модулей к системе лицензирования пакета программ ЛОГОС, а именно:

- подключение расчетных модулей к серверу лицензий через клиента, входящего в состав системы лицензирования;
- подключение пакетов программ к системе выдачи лицензий с помощью конфигурационного

файла с целью регистрации и выдачи лицензии для данного продукта;

- подключение приложений, имеющих графический интерфейс, к серверу лицензий с использованием графического клиента, входящего в состав системы лицензирования.

Состав системы лицензирования

Система лицензирования ЛОГОС состоит из следующих компонентов:

- сервер лицензий;
- набор клиентских библиотек;
- системы выдачи лицензий.

Общая схема работы сервера лицензии приведена на рис. 1.

Для внедрения нового программного продукта в систему лицензирования необходима интеграция клиентов сервера лицензий в подключаемые программные продукты, а также его внедрение в систему выдачи лицензий. Внедрение в сервер лицензий не требуется, сервер настраивается динамически на основе загруженных лицензий.

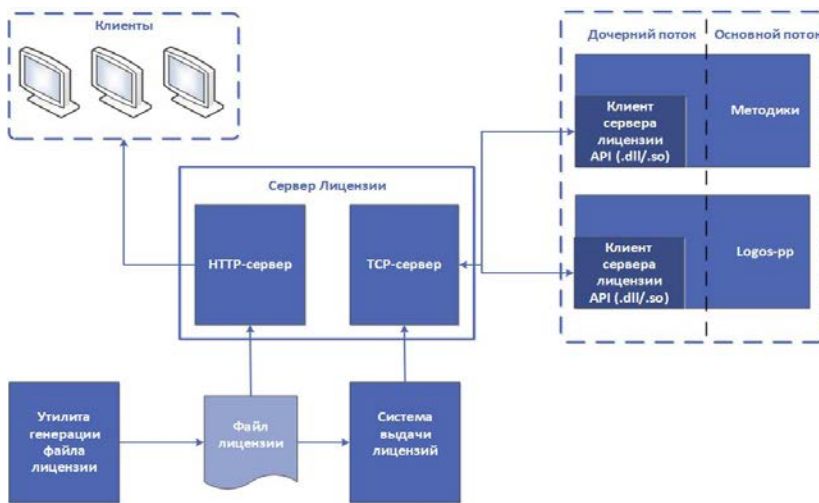


Рис. 1. Общая схема работы сервера лицензий

Базовый клиент для подключения к серверу лицензий

Базовый клиент инкапсулирует обмен данными по сети с сервером лицензий. Клиент разработан в виде динамической библиотеки на языке С, подключаемой к приложению с помощью добавления вызовов функций в исходных кодах приложений или других клиентов сервера лицензий. Клиент кросс-платформенный и работает под операционными системами Windows и Linux. Программный интерфейс базового клиента представлен в табл. 1.

Таблица 1

Программный интерфейс базового клиента

Название функции	Назначение функции
LIC_API int init(solver_description_t* solvers, size_t solvers_size, int& timeout)	Инициализация с сервером лицензий
LIC_API int step()	Запрос у сервера на продолжение работы клиента с текущей лицензией
LIC_API int reconnect ()	Переподключение к серверу лицензий после потери связи
LIC_API void finalize()	Уведомление сервера о завершении работы клиента
LIC_API disabled_features()	Получение списка запрещенных функциональных модулей
LIC_API feature_avail()	Проверка доступности функциональных модулей

Подключение расчетных модулей

Взаимодействие расчетного модуля с сервером лицензий осуществляется при помощи статической библиотеки license_client_starter. Данная библиотека разработана на основе базового клиента сервера лицензий и позволяет ограничивать количество запускаемых MPI-процессов.

При запуске расчетного модуля происходит вызов метода инициализации клиента сервера лицензий license_client_starter_init(), которому в качестве параметров подается имя расчетного модуля, а также его версия.

Во время выполнения данного метода копии библиотеки license_client_starter, каждая из которых принадлежит своему MPI-процессу расчетного модуля, обмениваются между собой данными.

В результате данного обмена формируется словарь, который хранит число MPI-процессов, на которых запущена задача.

Копия библиотеки license_client_starter, запущенная на MPI-процессе, ранг которого равен 0, передаст данную информацию через license_client_api серверу лицензий, который, в свою очередь, вернет клиенту информацию о возможности запуска расчета, таймаут опроса клиентом лицензионного сервера и уникальный идентификатор задачи.

Полученный ответ о возможности запуска расчета рассылается от процесса с нулевым рангом остальным. Если запуск невозможен, будет выведена информация о причине остановки расчета, метод license_client_starter_init() вернет отрицательный статус, что, в свою очередь, вызовет прекращение расчета.

В случае положительного ответа license_client_starter запустит дочерний поток, который через license_client_api будет отправлять серверу лицензий с определенным интервалом уведомления о том, что расчет продолжается. Сервер анализирует данные уведомления и отправляет клиенту сообщение о возможности продолжения работы. Если продолжение расчета невозможно, произойдет его остановка, также будет выведена информация о причине остановки.

При завершении работы расчетного модуля произойдет вызов метода завершения работы лицензионного клиента license_client_starter. Клиент завершает свою работу и через license_client_api уведомляет об этом сервер.

Схема взаимодействия расчетного модуля с сервером лицензий представлена на рис. 2.

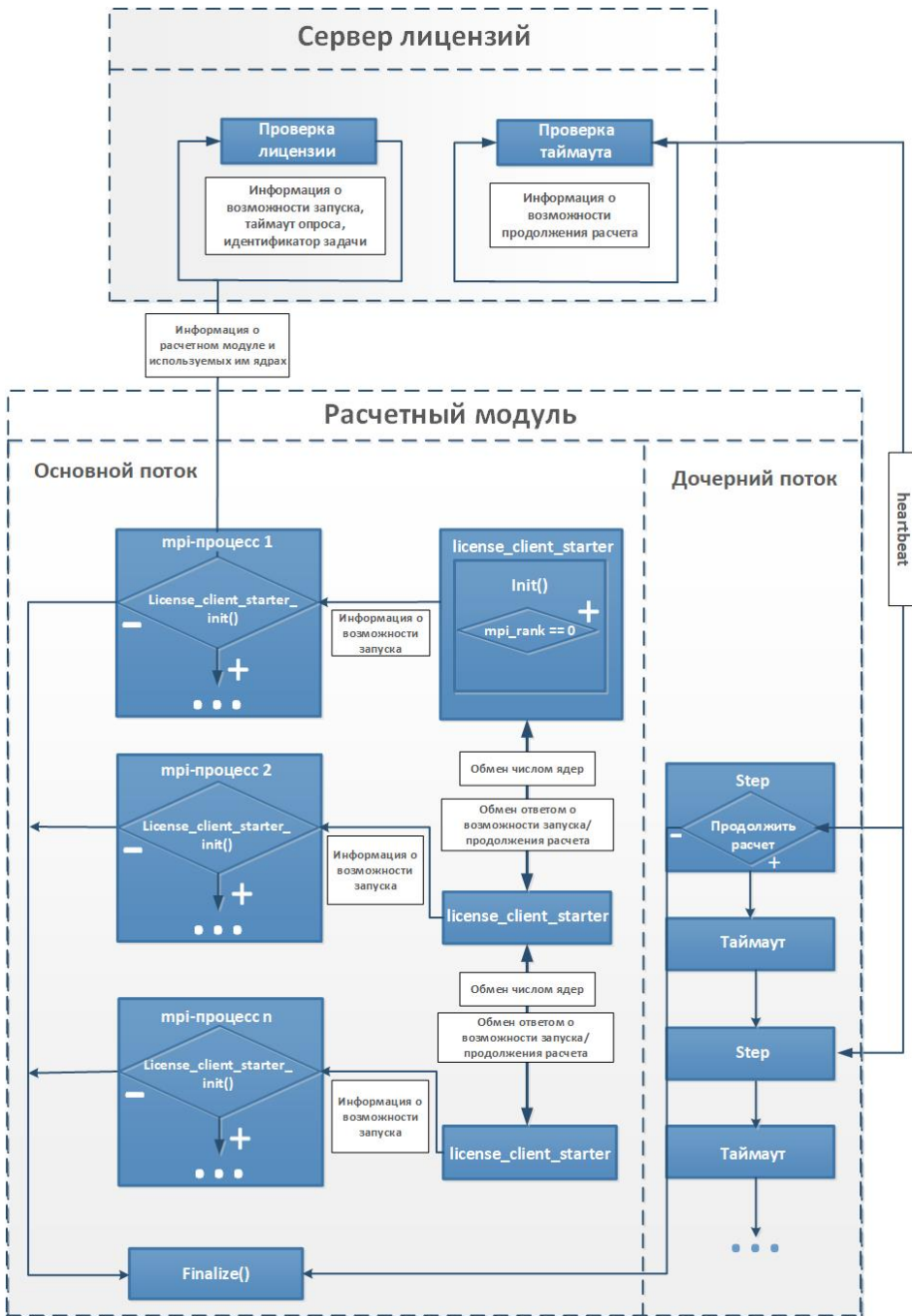


Рис. 2. Взаимодействие сервера лицензий с программным модулем

Механизм подключения

Для встраивания клиентской части сервера лицензий в расчетный модуль выполняется вызов методов `license_client_starter_init()` и `license_client_starter_finalize()` клиента сервера лицензий в момент инициализации и завершения работы расчетного модуля, как показано на рис. 3 и 4, соответственно.

```
#ifdef USE_LICENSE
    if (!license_client_starter_init("NYMPH", g_sVersion))
    {
        MPI_Finalize();
        exit(1);
    }
#endif
```

Рис. 3. Вызов функции `license_client_starter_init()`

```
#ifdef USE_LICENSE
    license_client_starter_finalize();
#endif
```

Рис. 4. Вызов функции `license_client_starter_finalize()`

Подключение приложений, имеющих графический интерфейс

Взаимодействие сервера лицензии с приложениями, имеющими графический интерфейс, реализованный на языке C++(Qt), осуществляется через библиотеку `usr_lic`. Данная библиотека разработана основе базового клиента сервера лицензий и предоставляет: графический диалоговый интерфейс для взаимодействия пользователя с сервером лицензий; возможность отправки уведомлений серверу лицензии о продолжении работы, с определенным интервалом в отдельном потоке основного приложения; возможность завершения работы приложения, в случае отправки сервером соответствующего уведомления. С помощью предоставляемого графического интерфейса пользователь при запуске приложения, при условии превышения параметров купленной лицензии, получает диалоговое окно с уведомлением. В данном диалоговом окне пользователь может выбрать, перепроверить лицензию или закрыть приложение. При выборе пользователем перепроверки лицензии инициируется повторная проверка лицензии через сервер; если занятые ресурсы купленной лицензии освободились, то происходит запуск приложения, а в противном случае снова появляется диалоговое уведомление о превышении лицензии. Взаимодействие приложения с сервером лицензии показано на рис. 5.

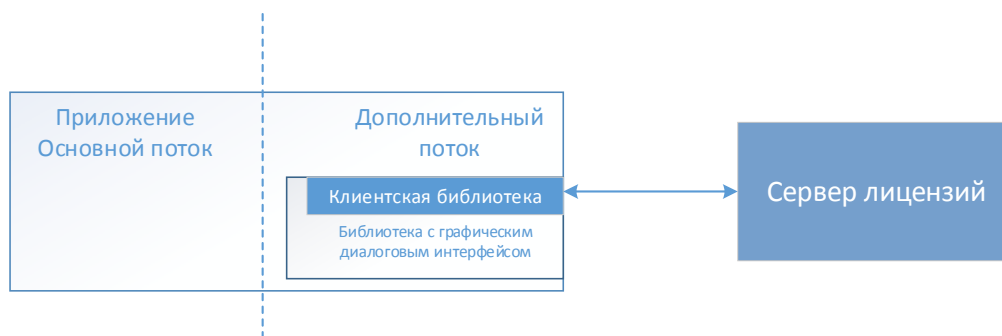


Рис. 5. Схема взаимодействия сервера лицензии с графическими приложениями

Программный Интерфейс для реализации взаимодействия между клиентом и графическим приложением методами, указанными в табл. 2.

Подключение к системе выдачи лицензий

Система выдачи и учета лицензий ЛОГОС служит для генерации и хранения файлов лицензий. Система представляет из себя веб-приложение, запускаемое в веб-браузере пользователя. Система позволяет создавать и редактировать лицензионные файлы, хранить историю выдачи лицензионных файлов пользователям, а также имеет разграничение прав на выдачу лицензий с помощью LDAP-аутентификации.

Интерфейс пользователя системы выдачи и учета лицензий ЛОГОС показан на рис. 6.

Для возможности использования других пакетов программ в системе лицензирования ЛОГОС, система выдачи лицензий адаптирована к подключению новых компонент и необходимым для их функционирования параметрам. Для реализации этого функционала был разработан формат конфигурационных файлов, который может описывать существующие и будущие продукты с любыми конфигурациями, а также добавлен модуль чтения конфигурационных файлов в этом формате.

Таблица 2

Программный интерфейс графического клиента

Название функции	Назначение функции
<code>static void Initialize(const char *mname)</code>	Инициализация переменных класса, необходимых для дальнейшей работы, <code>mname</code> – имя приложения
<code>bool DoWork(bool bGUIApplicationMode)</code>	Осуществляет инициализацию работы приложения с сервером лицензий и выполняет запуск потока и выполнение цикла работы клиента в потоке
<code>static LicThreadClient* GetClient()</code>	Возвращает указатель на единственный экземпляр класса
<code>static void Finalize()</code>	Завершение работы клиента

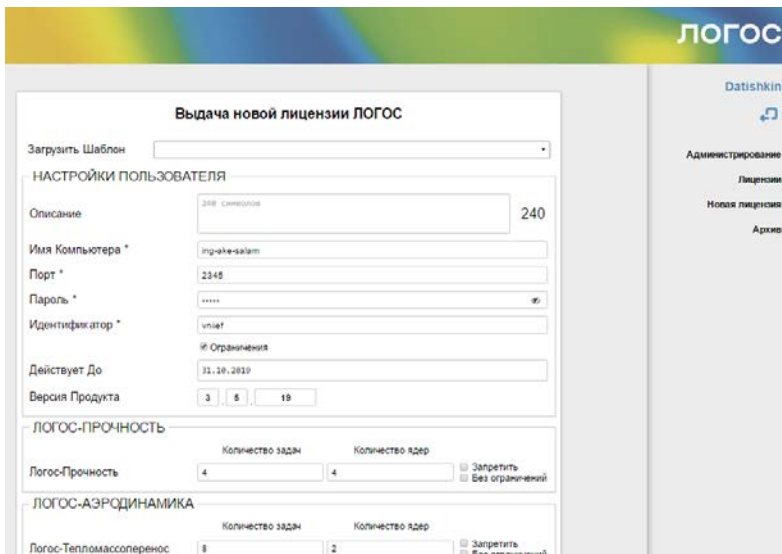


Рис. 6. Интерфейс пользователя системы выдачи и учета лицензий ЛОГОС

Общий формат конфигурационного файла имеет две секции: `project_name` и `methodics`. `Project_name` содержит название проекта для различных языков интерфейса. В настоящий момент доступны русский и английский языки. Данная секция представлена на рис. 7.

```
{
  "project_name": {
    "ru": "Логос Базовый",
    "en": "Logos Base"
  },
}
```

Рис. 7. Секция `project_name` конфигурационного файла системы выдачи лицензий

Секция `methodics` является массивом, содержащим объекты описания модулей. Интерфейс каждого объекта представлен на рис. 8.

Описание всех полей лицензии представлено в табл. 3.

Таблица 3

Описание полей лицензии

key	Символьное имя группы методики (например: <code>heat</code> , <code>sa</code> , <code>pp</code>)
<code>name</code>	Объект текстового названия для различных языков
<code>modules</code>	Массив объектов модулей, содержащихся в группе
<code>permissions</code>	Массив объектов доступных ограничений для модуля
<code>id</code>	Строковое описание ограничения для модуля, предоставляемое разработчиком
<code>type</code>	Текстовое описание типа ограничения, в настоящий момент поддерживается три типа: <code>int</code> , <code>double</code> , <code>bool</code>
<code>value</code>	Значение по умолчанию для ограничения модуля, в случае если тип ограничения <code>int</code> или <code>double</code> , он имеет вид числовой строки, если ограничение имеет тип <code>bool</code> , может принимать два булевых значения: <code>true</code> или <code>false</code>

```
{
  "methodics": [
    {
      "key": "string",
      "name": {
        "ru": "string",
        "en": "string"
      },
      "modules": [
        {
          "key": "string",
          "name": {
            "ru": "string",
            "en": "string"
          },
          "permissions": [
            {
              "name": {
                "ru": "string",
                "en": "string"
              },
              "id": "string",
              "type": "string",
              "value": "boolean | string"
            }
          ]
        }
      ]
    }
  ]
}
```

Рис. 8. Интерфейс описания методик в конфигурационном файле лицензий

Для добавления нового типа конфигурации необходимо составить конфигурационный файл и поместить его в папку `configs/license_templates`. После этого необходимо перезапустить систему выдачи лицензий. Если конфигурация написана без ошибок, то в системе выдачи появится новый тип выдаваемой лицензии, например, как изображено на рис. 9.

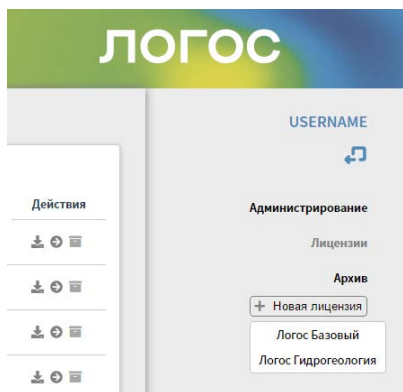


Рис. 9. Список добавленных шаблонов для выдачи лицензий

Также конфигурация может содержать в себе несколько числовых и булевых типов ограничений. Ее вид в системе выдачи лицензий изображен на рис. 10.

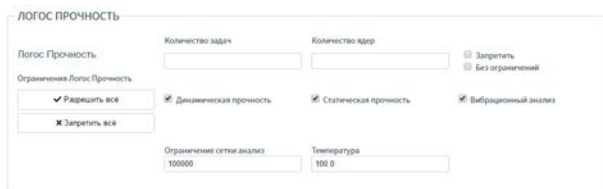


Рис. 10. Отображение конфигурации ЛОГОС ПРОЧНОСТЬ в системе выдачи лицензий

Конфигурируемая утилита настройки подключения к серверу лицензий

Утилита настройки параметров подключения к серверу лицензий предназначена для указания параметров сервера лицензий как для локального расчета, так и для расчета, проводимого на супер-ЭВМ. Для предоставления возможности настройки разных программных продуктов без изменения исходных кодов утилиты разработан конфигурационный файл, представленный на рис. 11.

```
{
  "name": { "ru": "Логос Гидрогеология", "en": "Logos Hydrogeology" },
  "id": "logos_hg",
  "config_file": "hg_lic_server_settings_v1.1.ini"
}
```

Рис. 11. Пример конфигурационного файла утилиты

Конфигурационный файл содержит в себе следующие поля:

- «name» – имя пакета программ;
- «id» – идентификатор пакета программ в системе лицензирования «Логос»;
- «config_file» – имя файла для хранения настроек подключения.

Внешний вид утилиты настройки параметров подключения к серверу лицензий для разных продуктов представлен на рис. 12.

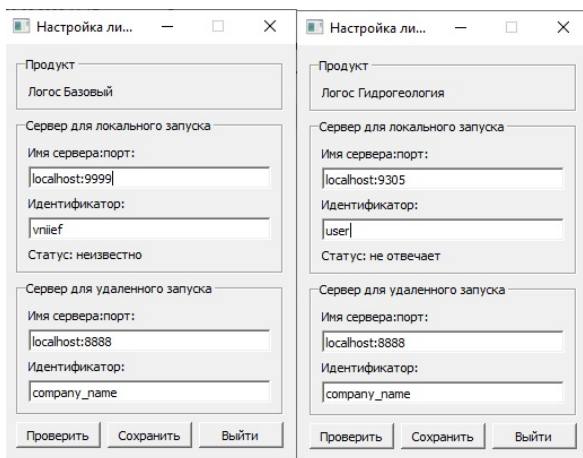


Рис. 12. Внешний вид утилиты настройки параметров подключения к серверу лицензий

Заключение

Разработанная методика подключения программных модулей позволяет гибко подключать к системе лицензирования ЛОГОС различные пакеты программ благодаря использованию набора различных клиентов и механизма конфигурационных файлов. Методика не требует значительного изменения программного кода подключаемых программных модулей.

Система лицензирования была внедрена в пакеты программ ЛОГОС базовый и ЛОГОС ГИДРОГЕОЛОГИЯ. Также возможно внедрение системы лицензирования в другие программные продукты, разрабатываемые в рамках развития ЛОГОС.

Список литературы

1. Пакет программ ЛОГОС. <http://logos.vniief.ru/products/logos>.