DOI 10.37882/2223–2966.2020.04–2.09

АВТОМАТИЗАЦИЯ КОНФИГУРИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ НАЗЕМНЫХ СТАНЦИЙ СИСТЕМЫ СПУТНИКОВОЙ СВЯЗИ

AUTOMATED CONFIGURATION OF THE SATELLITE COMMUNICATION SYSTEM GROUND STATIONS SOFTWARE

N. Ignatov
N. Sechkina

Summary. The article is devoted to the automating software configuration for deploy to customer facilities in the rocket and space industry of the Russian Federation. An overview of existing configuration management tools is provided. Automated software configuration method is proposed within the existing enterprise infrastructure. A software system has been developed that implements the proposed approach. The scientific novelty of the study consists in the possibility of transferring and subsequent operation of the configuration system on various software platforms, including specialized military operating systems for the Russian Federation.

Keywords: software configuration, ground station, operating facility, configuration parameters, configuration template, continuous integration server.

Игнатов Николай Анатольевич
К.т.н., доцент, Московский авиационный институт (национальный исследовательский университет)
nick_ignatov@mail.ru
Сечкина Наталья Сергеевна
Московский авиационный институт (национальный исследовательский университет); инженер, АО «Научно-исследовательский институт точных приборов» (г. Москва)
sechkina.natali@yandex.ru

Аннотация. Статья посвящена вопросам автоматизации конфигурирования программного обеспечения при поставке на объекты заказчика в ракетно-космической отрасли Российской Федерации. Проводится обзор существующих средств конфигурационного управления. Предложен метод конфигурирования программного обеспечения в рамках существующей инфраструктуры предприятия. Разработана программная система, реализующая предложенный подход. Научная новизна исследования состоит в возможности переноса и последующей работы системы конфигурирования на различных программных платформах, в том числе в специализированных операционных системах военного назначения для Российской Федерации.

Ключевые слова: конфигурация программного обеспечения, наземная станция, объект эксплуатации, конфигурационные параметры, шаблон конфигурации, сервер непрерывной интеграции.

Введение

Современные программные системы, разрабатываемые для нужд ракетно-космической отрасли Российской Федерации, характеризуются повышенной степенью сложности, обусловленной высокой наукоемкостью программного обеспечения. Для сохранения лидирующих позиций в области создания программных продуктов в ракетно-космической промышленности необходимо разрабатывать, непрерывно совершенствовать и поставлять в кратчайшие сроки надежные и качественные программные системы, способные удовлетворить потребности заказчика. Проблемы конфигурации таких программных продуктов выходят на первый план.

Конфигурация программного обеспечения — совокупность настроек задаваемая пользователем [1, с.90]. Автоматизация конфигурирования программного обеспечения проводилась на примере деятельности АО «НИИ ТП». В настоящее время конфигурированием программных систем в подразделении АО «НИИ ТП» занимается уполномоченное лицо, которое выполняет настройку каждого программного компонента для отдельной ЭВМ на объекте заказчика в ручном режиме. Отсутствие автоматизации увеличивает время развертывания программного обеспечения: затрачивается время на поиск и устранения ошибок конфигурирования, а также на редактирование параметров во множестве конфигурационных файлов [1, с.90]. Такой неавтоматизированный процесс снижает качество программного

Рис. 1. Схема работы системы управления конфигурациями

продукта, увеличивает время поставки и снижает его надежность, что в общем случае критично для систем военного назначения.

Постановка задачи

Дано:

Программный комплекс конфигурируется и развертывается на территориально-распределенных наземных станциях с ЭВМ. Программный комплекс состоит из программных компонент. Каждый программный компонент имеет конфигурационных параметров в конфигурационных файлах, где $i \in j, i = \overline{1, K}; j = \overline{1, L}.$

Требуется:

Автоматизировать процесс задания конфигурационных параметров программных компонент для различных объектов эксплуатации (территориально-распределенных наземных станций) в конфигурационных файлах.

Обзор существующих средств управления конфигурацией

На данный момент существует несколько готовых систем, способных автоматизировано настраивать и поставлять программное обеспечение, к ним относятся CFEngine, SaltStack, Chef, Puppet и Ansible. Общая схема работы перечисленных инструментов представлена на рисунке 1.

В центральном хранилище расположены инструкции, в которых содержится описание того, что необходимо сделать на узле, например: установить и настроить определенное программное обеспечение, запустить службу, выполнить удаление библиотеки, изменить конфигурационные параметры программных компонент. Эти инструкции переводятся на узел трансляции, после чего они передаются (или забираются) на управляемые узлы и превращаются в набор исполняемых команд при помощи компонента внедрения [2].

При анализе данных систем выявлено, что перечисленные средства конфигурационного управления не являются полностью кроссплатформенными и не поддерживают российскую операционную систему МСВС, эксплуатируемую на объектах поставки. В связи с этим использовать перечисленные системы непосредственно на наземной станции невозможно. Кроме того, основной принцип работы этих средств — использование стратегии непрерывной поставки программного обеспечения. Таким образом, при использовании систем управления конфигурацией переход между процессами компиляции, сборки, конфигурирования, тестирования и развертывания является непрерывным. На объектах военного назначения, где используется защищенное сетевое соединение, удаленное использование этих средств конфигурационного управления невозможно. Поэтому возникает необходимость в создании собственной кроссплатформенной системы конфигурирования и развертывания ПО для предприятия и обслуживаемых им объектов [3, с. 114].

Принцип работы программы конфигурирования

Предлагаемый метод конфигурирования программной системы основан на применении шаблонов конфигурации. Шаблон представляет собой размеченный по определенным правилам файл, в котором используются переменные, функции, условия и циклы. Конфигурационные шаблоны имеют то же расширение, что и конфигурационные файлы — это *.xml, *.reg, *.ini и *.conf. Примеры конфигурационных шаблонов представлены на рисунке 2 (XML формат) и на рисунке 3 (файл реестра Windows).

```
<?xml version="1.0" encoding="utf-8"?>
<config>
  <key name="DB">
    <key name="Server" value="{$.get("db_name")}"/>
    <key name="User" value="{$.get("user_name")}"/>
    <key name="Password" value="{$.get("pass")}"/>
  </key>
  <key name="General">
    <key name="Templates" {if ($.flag("CONF_GONEC")) 'value="tmpl_gonec.xml"/>'; else if ($.flag("CONF_RODNIK")) 'value="tmpl_rodnik.xml"/>';}
    <link name="PZEditor" to="Editors/1"/>
  </key>
  <key name="Editors">
    <key name="1" value="plz"/>
  </key>
</config>
```

Рис. 2. Пример шаблона для файла формата XML

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\NIITP\event]
"User"="{$.get("user_name")}"
"Password"="{$.get("pass")}"
"DBName"="{$.get("db_name")}"
"Interval"="60"
"MaxNumber"="1000"
"NumPK"="1"
"TypePK"="2"
"Priority"="1"
"Width"="654"
"Height"="294"
"X"="291"
"Y"="288"
"NS"="10"
```

Рис. 3. Пример шаблона для файла формата REG

В этих шаблонах вместо заранее заданных значений ключей настроек используются переменные, заключенные в специальные разделители вида: “{}”. Программа конфигурирования (далее «Конфигуратор») разработана на языке программирования C++ с использованием межплатформенной библиотеки Qt версии 4.8.6. В качестве интерпретатора языка шаблонов в программе используется QtScript, базирующийся на стандарте ECMA-262 (так же известном как JavaScript 2.0 или Jscript.NET).

В приведенных примерах используется функция get с аргументом — именем переменной, значение которой хранится в базе данных проектов отдела разработки ПО АО «НИИ ТП». Данная функция определена в программном коде Конфигуратора на языке C++ и вызывается в шаблоне при его обработке интерпретатором QtScript. Прототип этой функции следующий:

QString get (const QString& val);

Функция get возвращает значение типа QString, которое представляет собой значение переменной val, извлеченное из базы данных для заданного объекта эксплуатации.

В шаблоне, представленном на рисунке 2, также используется конструкция ветвления типа if…else для выбора между двумя альтернативами — значениями ключа «Templates». Функция flag предназначена для проверки наличия настроек соответствующей наземной станции в базе данных проектов и также определена в программном коде Конфигуратора на языке C++.

Прототип этой функции следующий:

bool flag (const QString& val);

В приведенном примере в функцию передается значение переменной val равное «CONF_GONEC» и «CONF_RODNIK». В случае если на наземной станции установ-

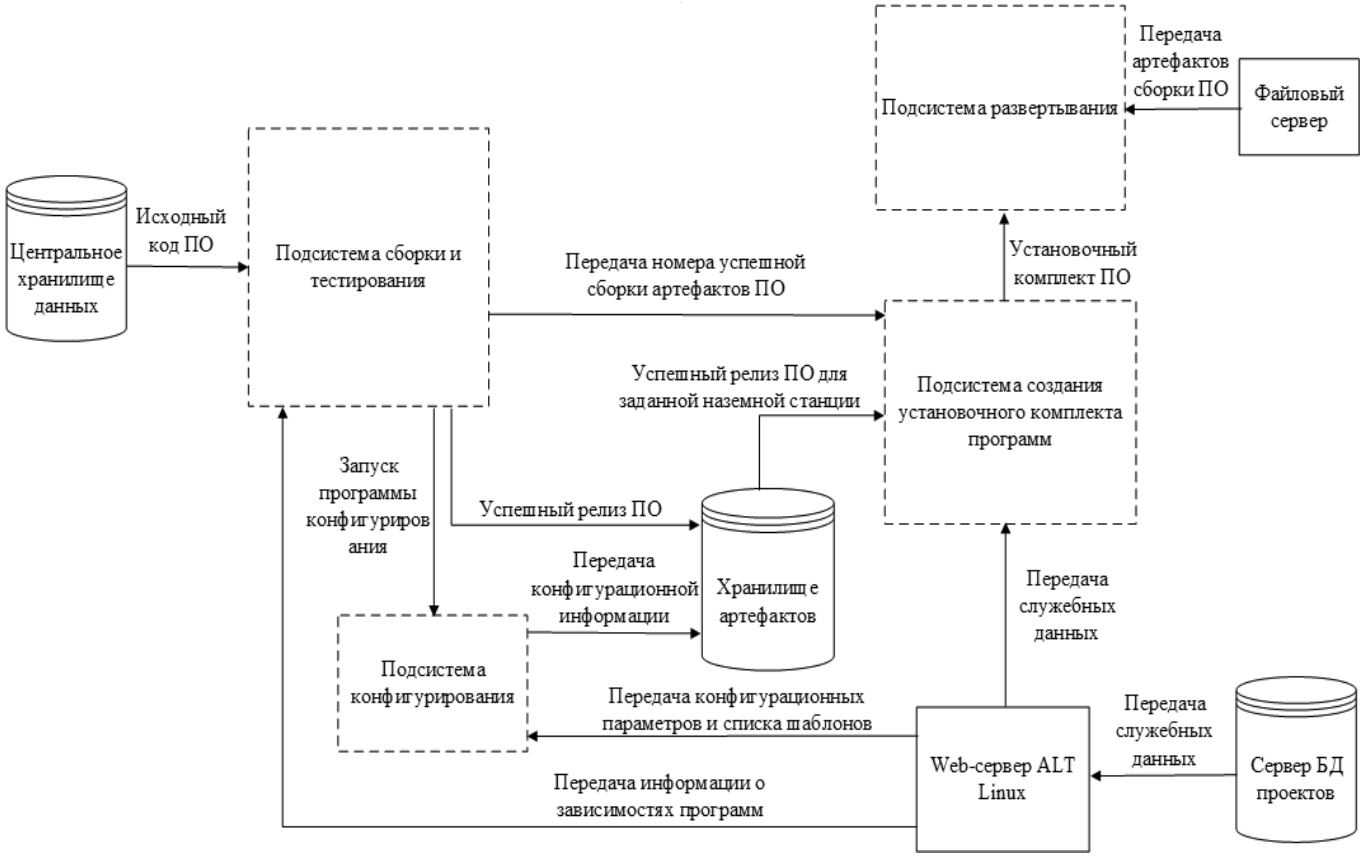

Рис. 4. Общая схема функционирования системы конфигурирования

лена настройка «CONF_GONEC», то будет использовано значение ключа «Templates» равное «tmpl_gonec.xml», если же на наземной станции установлена конфигурация CONF_RODNIK, то значение ключа «Templates» будет равно «tmpl_rodnik.xml».

Описанные выше функции являются методами объекта $, который представляет собой объект сценария, зарегистрированный в коде программы конфигурирования.

Общая схема работы системы конфигурирования в рамках существующей инфраструктуры предприятия представлена на рисунке 4.

При фиксации изменений программистом в ветке разработки происходит запуск сборки программной системы на сервере непрерывной интеграции (в качестве сервера непрерывной интеграции используется Jenkins). Ведущий сервер сборки извлекает из хранилища системы управления версиями исходный код программной системы для сборки, а также шаблоны конфигурации, предварительно созданные разработчиком и помещенные в центральное хранилище системы управления версиями. Машина сборки также извлекает из базы данных проектов информацию о зависимостях программ. На этапе сборки на сборочной машине функционирует программа конфигурирования, настроенная на работу в качестве шага сборки сервера непрерывной интеграции.

В файле настроек Конфигуратора задается адрес web-сервера, с помощью API которого взаимодействует программа конфигурирования и база данных проектов. Номер объекта, для которого конфигурируется программная система, передается Конфигуратору в качестве переменной окружения. Этот номер необходим для извлечения конфигурационных параметров и списка шаблонов конфигурации из базы данных проектов для определенного объекта эксплуатации. Алгоритм работы системы конфигурирования представлен на рисунке 5 в виде UML-диаграммы активностей.

На выходе Конфигуратор генерирует из шаблонов конфигурационные файлы, которые включаются в архив (инсталляционный комплект) программной системы вместе с набором программ и служебной информацией. Далее установочный комплект доставляется уполномоченными органами на файловый сервер наземной станции, где происходит непосредственная установка

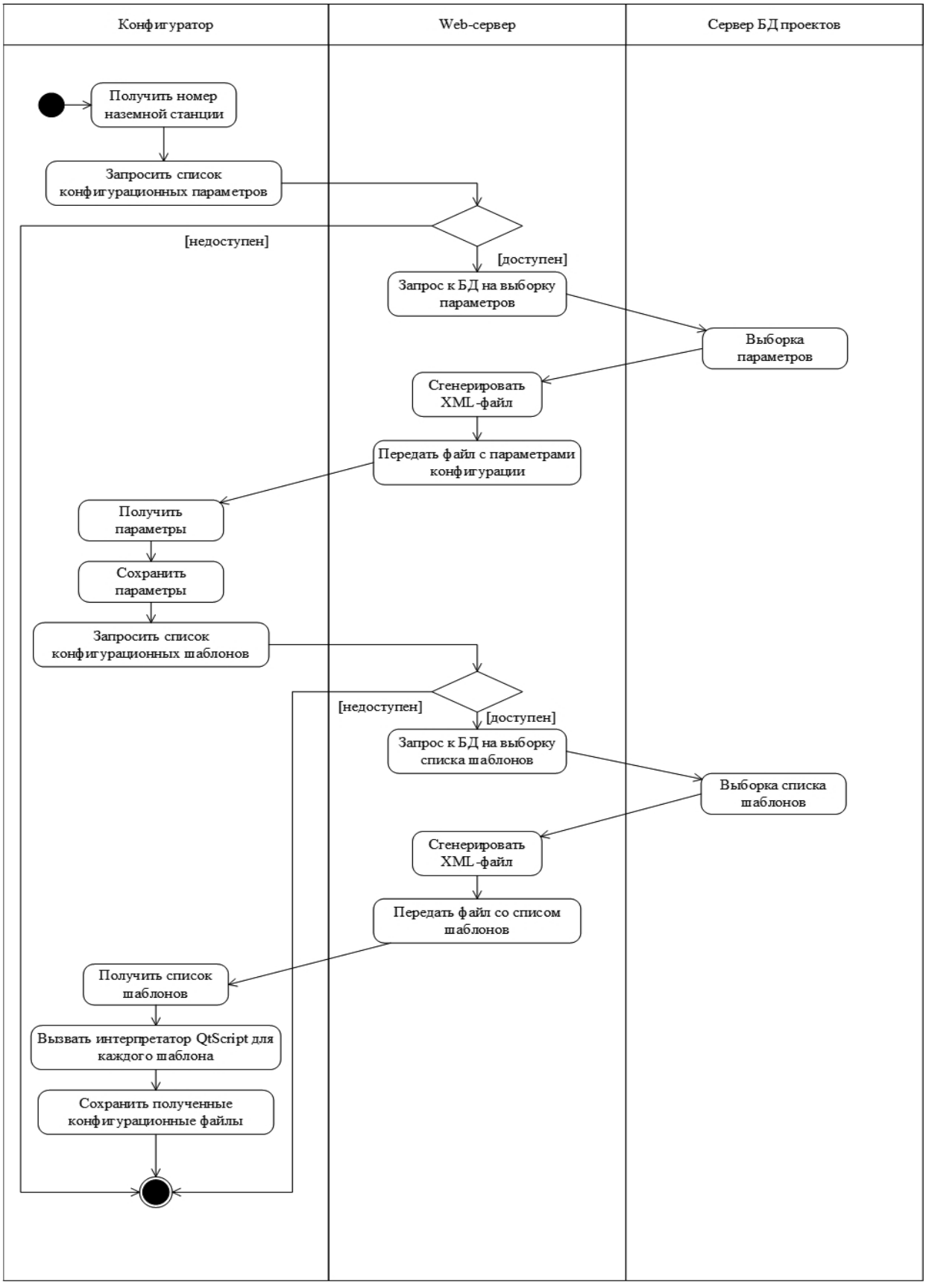

Рис. 5. Алгоритм работы системы конфигурирования

Таблица 1. Результаты применения системы развертывания

Действие	Ручное время развертывания (чч: мм: сс)	Время автоматизированного развертывания (чч: мм: сс)
Установка конфигурационных данных программной системы на вычислительные узлы наземной станции	02:30:44	00:05:08
Установка артефактов сборки ПО на вычислительные узлы наземной станции по рабочим директориям	01:20:33	00:00:09
Установка и запуск необходимых служб на вычислительных узлах наземной станции	00:08:46	00:00:05
Итого	04:00:03	00:05:22

программ и их конфигурации на узлы объекта системой развертывания в автоматизированном режиме, разработанной в подразделении АО «НИИ ТП» (подсистема развертывания).

Анализ результатов применения системы развертывания

Анализ результатов применения системы развертывания проводился на основе следующей выборки:

Количество конфигурационных файлов: 95 файлов;
количество артефактов сборки ПО: 305 файлов;
количество служб: 5 файлов.

Результаты применения системы развертывания представлены в таблице 1.

В таблице показано среднее время, затрачиваемое на развертывание программной системы в ручном и автоматизированном режимах. Ручное время развертывания указано в случае работы одного специалиста отдела эксплуатации программного обеспечения по установке и настройке программной системы для одной наземной станции. Проанализировав таблицу представленную выше, можно увидеть, что в случае применения системы развертывания на объекте эксплуатации удалось сократить время установки и настройки программного комплекса с 4 часов в ручном режиме до 5 минут в случае применения системы автоматизации.

Заключение

При изучении существующих средств конфигурационного управления выявлено, что их использование на объектах военного назначения невозможно. В связи с этим разработана собственная программная система конфигурирования, которая является кроссплатформенной и легко внедряется в существующую инфраструктуру предприятия. Использование системы конфигурирование позволило снизить административные расходы при поставке программной системы на наземные станции, уменьшить ее время развертывания и сократить количество конфигурационных ошибок программных компонент.

ЛИТЕРАТУРА

1. 18-я Международная конференция «Авиация и космонавтика — 2019» (Москва, МАИ, 18–22 ноября 2019 г.): тезисы / (Московский авиационный институт). Москва: Логотип, 2019. 605 с.
2. Житинский С., Чистяков А. Системы управления конфигурацией. Чем отличаются Chef, SaltStack, Puppet, CFEngine и Ansible? // Системный администратор. 2014. № 6(139). С. 35–39.
3. Восьмая Всероссийская научно-техническая конференция «Безопасные информационные технологии» (Москва, 6–7 декабря 2017 г.): сборник трудов конференции / под общ. ред. М. А. Басараба. Москва: МГТУ им. Н. Э. Баумана, 2017. 541 с.

4. Амиров А.Ж., Герхардт Э., Хон М. В. Особенности процесса развертывания программного обеспечения в условиях интенсивной разработки // Молодой ученый. 2016. № 9(113). С. 44–47.
5. Воробьев О.А., Павлов Л. С. Автоматизация развертывания компонент распределенного приложения современными средствами управления конфигурацией. // Молодой ученый. 2016. № 13(117). С. 306–311.
6. Брусакова И.А., Раимова С. Т. Система конфигурационного управления — инновационный инструмент сопровождения процесса разработки программного обеспечения (ПО) в авиационной промышленности // Инновации. № 8 (142). С. 96–100.
7. Хамбл Д., Фарли Д. Непрерывное развертывание ПО: автоматизация процессов сборки, тестирования и внедрения новых версий программ.: Пер. с англ. — М.: ООО "И. Д. Вильямс", 2011. — 432 с.
8. Уайт Б. А. Управление конфигурацией программных средств. Практическое руководство по Rational ClearCase: Пер. с англ. — М.: ДМК Пресс, 200. — 272 с.
9. Липаев, В. В. Программная инженерия. Методологические основы: Учеб. / В. В. Липаев; Гос. ун-т — Высшая школа экономики. — М.: ТЕИС, 2006. — 608 с.
10. Вольф Э. Continuous delivery. Практика непрерывных апдейтов. — СПб.: Питер, 2018. — 320 с.

Журнал «Современная наука: актуальные проблемы теории и практики»

Московский авиационный институт