НАУЧНОЕ ИЗДАНИЕ МГТУ ИМ. Н. Э. БАУМАНА
НАУКА и ОБРАЗОВАНИЕ
Эл № ФС77 - 48211. ISSN 1994-0408
приложение

Применение системы контроля версий GitLab для обучения программированию

10, октябрь 2016
DOI: 10.7463/1016.0848154

Сидякин И. М.[1,*]

УДК 378.146

[1]МГТУ им. Баумана, Москва, Россия
[*]ivan.sidyakin@gmail.com

Практическая часть учебных курсов по программированию, как правило, содержит задания, которые выполняются учащимися самостоятельно на лабораторных занятиях в классе или дома. Для проверки таких заданий удобно использовать системы дистанционного обучения. Обычно применяются специализированные системы поддержки обучения, имеющие массу дополнительных функций, включая возможность публикации учебных материалов, обмен сообщениями между учащимися и преподавателем, контроль знаний учащихся, подсчет статистических показателей и прочее [3].

В этой работе представлены рекомендации по применению в учебных целях системы GitLab [1], которая используется профессиональными разработчиками программного обеспечения *контроля версий*. Система контроля версий помогает организовать совместную работу группы разработчиков над общим программным проектом, обеспечивает хранение программного кода в распределенном хранилище (репозитории), регистрацию изменений в коде, возможность отмены этих изменений и возврата к прошлым версиям программного кода.

Кроме этих основных функций, GitLab предлагает дополнительные инструменты, аналогичные тем, которые используют системы дистанционного обучения. Например, GitLab имеет встроенную *систему отслеживания ошибок* [8], которую удобно использовать для обмена замечаниями, вопросами и ответами, касающимися исполнения заданий. Система контроля версий GitLab имеет средства документирования, которые можно использовать для публикации учебных материалов, а также поддерживает набор функций Web-API, который позволяет автоматизировать процедуры управления пользователями и проектами внутри системы.

Доступ к ресурсам GitLab невозможен без авторизации. Это, в частности, означает что преподаватель всегда может посмотреть кто и когда внес изменения в ответы на задания, а также установить персональные права доступа к информационным материалам, сообщениям и программному коду для учащихся.

Следует подчеркнуть что в этой статье речь идет о курсах по программированию, в которых ответы на задания, требуемые от учащихся это исходный код программ написанный на каком либо из языков программирования, например, C, C++, Java.

Резюмируя сказанное, перечислим основные свойства системы контроля версий GitLab, которые делают возможным ее превращение в инструмент обучения программированию:

1. Автоматизация передачи исходного кода практических заданий по программированию между компьютерами преподавателя и учащихся.
2. Обмен сообщениями между преподавателем и учащимися в формате системы отслеживания ошибок. Преподаватель может общаться персонально с каждым учащимся или публиковать доступные для всех слушателей курса комментарии.
3. Публикация заданий, материалов лекций и других учебных ресурсов на Wiki страницах проектов GitLab.
4. Использование Web-API для автоматизации регистрации пользователей, создания шаблонов проектов для выполнения практических заданий, синхронизации репозитория содержащего выполненные задания на компьютерах учащихся и преподавателя.

Заметим, что в системе отсутствует важная деталь - журнал успеваемости. Нет средств для записи оценок за выполненные задания. Журнал можно, например, вести в одном из онлайн сервисов для работы с документами, например, Google Docs или iCloud.

Перейдем к вопросу практической реализации системы контроля версий. GitLab, это надстройка над системой контроля версий Git [2]. На странице википедии https://ru.wikipedia.org/wiki/Git вы найдете краткое описание Git и ссылки на различные ресурсы, необходимые для детального изучения этой системы. Git имеет длинную историю и очень большой набор свойств и возможностей. К счастью, для того чтобы начать работу с Git, достаточно уяснить основные принципы работы и освоить несколько простых операций. В этой статье не ставится цель изучения правил работы с Git, поэтому, для поиска ответов на все относящиеся к этой теме вопросы, воспользуйтесь приведенной выше ссылкой.

Git можно использовать в разных операционных системах. В этой статье, в примерах, используется командная строка Linux Ubuntu [4]. Git должен быть установлен на компьютерах всех участников курса по программированию: учащихся и преподавателя. ОС Linux выбрана также по той причине, что в этой операционной системе проще написать командные скрипты [9], облегчающие преподавателю процесс проверки заданий.

Git это основа системы контроля версий, движок, обеспечивающий совместный доступ пользователей к исходному программному коду и регистрацию изменений вносимых

в этот программный код пользователями. GitLab - надстройка над Git, которая добавляет в систему полезные инструменты позволяющие организовать работу над проектом или группами проектов в команде. GitLab это веб-приложение которое устанавливается на сервере в сети и должно быть доступно на компьютерах всех участников курса. Если отсутствуют противопоказания в требованиях к безопасности, сервер GitLab желательно разместить в сети Интернет, так чтобы он был доступен пользователям в учебных лабораториях и дома.

Установка и администрирование GitLab не должны вызвать осложнений. Дистрибутив и инструкции по установке бесплатно распространяемой версии GitLab Community Edition расположены по этой ссылке: https://about.gitlab.com/downloads/. Следует отметить, что именно эта версия программы бесплатна, и предназначена для установки на ваш собственный сервер.

Перейдем к вопросу практической реализации системы поддержки обучения на основе GitLab. В этой задаче следует выделить три основные составляющие: настройка учебного курса на сервере GiLab, настройка компьютера преподавателя и настройка компьютера учащегося.

Настройку учебного курса на сервере GitLab следует начать с регистрации пользователей. Регистрация преподавателя и учащихся не имеет принципиальных отличий. Каждый получает логин и пароль для доступа к ресурсам сервера. Вместо логина допускается использовать адрес электронной почты. Для доступа к исходному коду проектов на сервере GitLab рекомендуется использовать протокол SSH и ключи SSH. Настройка способа авторизации с помощью ключей подробно описана в документации GitLab.

Следующий шаг это создание учебного проекта, который будет содержать ресурсы необходимые для того, чтобы учащиеся могли начать работу над своими заданиями. На этом этапе преподавателю достаточно авторизоваться на сайте сервера GitLab и создать новый пустой проект.

Рассмотрим случай, когда практический раздел курса по программированию содержит набор независимых или же связанных по смыслу заданий (лабораторных работ), которые обязан выполнить каждый из учащихся. Все эти задания рекомендуется объединить в один проект GitLab и разработать для них структуру каталогов. Например, для каждого задания можно создать отдельный каталог, предназначенный для хранения файлов исходного кода и других ресурсов, относящихся к этому заданию. Если для выполнения заданий требуются готовые фрагменты исходного кода или файлы данных, их можно заранее добавить в структуру каталогов проекта, На рисунках 1 и 2 показан пример проекта учебного курса, состоящий из восьми лабораторных работ.

Добавить к проекту каталоги и файлы можно в ручном режиме с помощью пользовательского интерфейса веб-приложения GitLab, однако гораздо проще создать структуру каталогов проекта и заполнить ее необходимыми для работы файлами на компьютере преподавателя. Затем добавить созданную структуру в проект GitLab. Подробное описание этой процедуры имеется в документации GitLab.

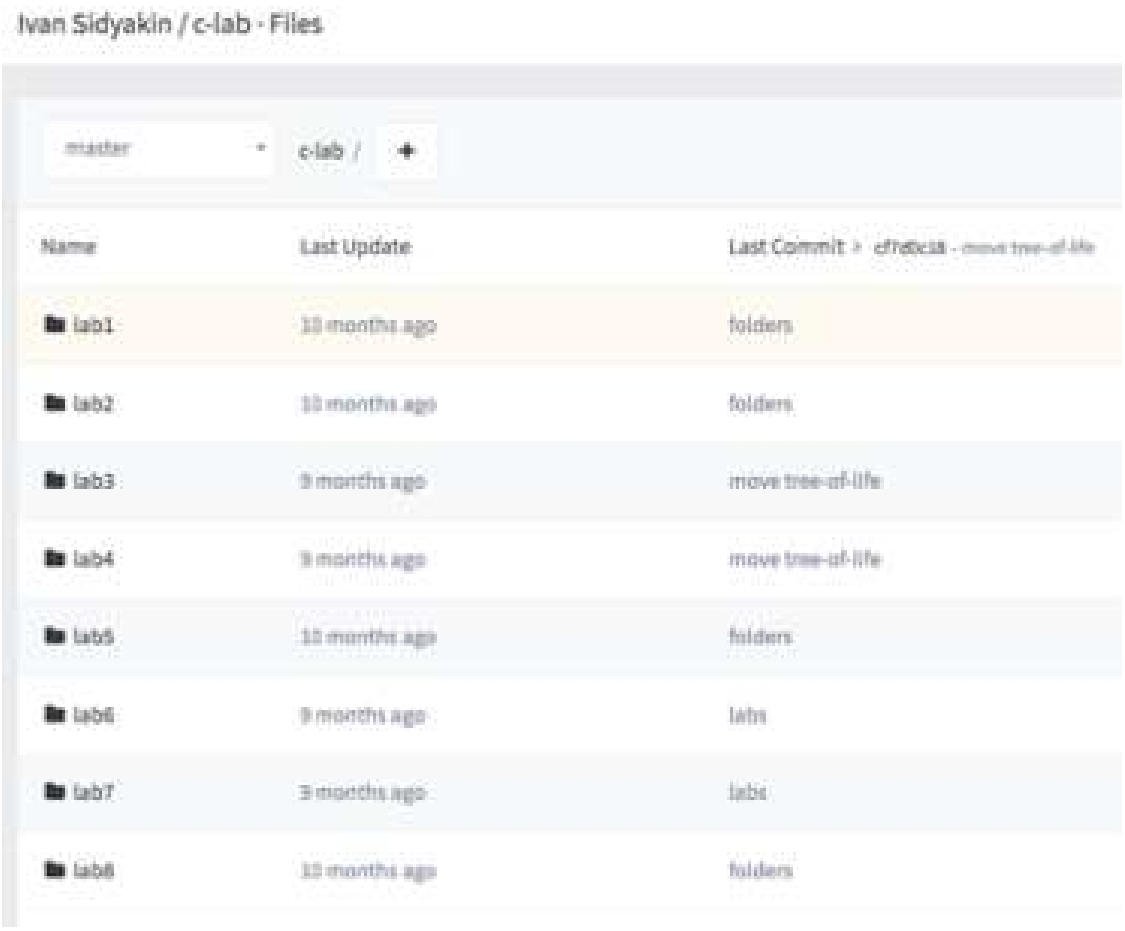

Рисунок 1. Структура каталогов учебного проекта в GitLab

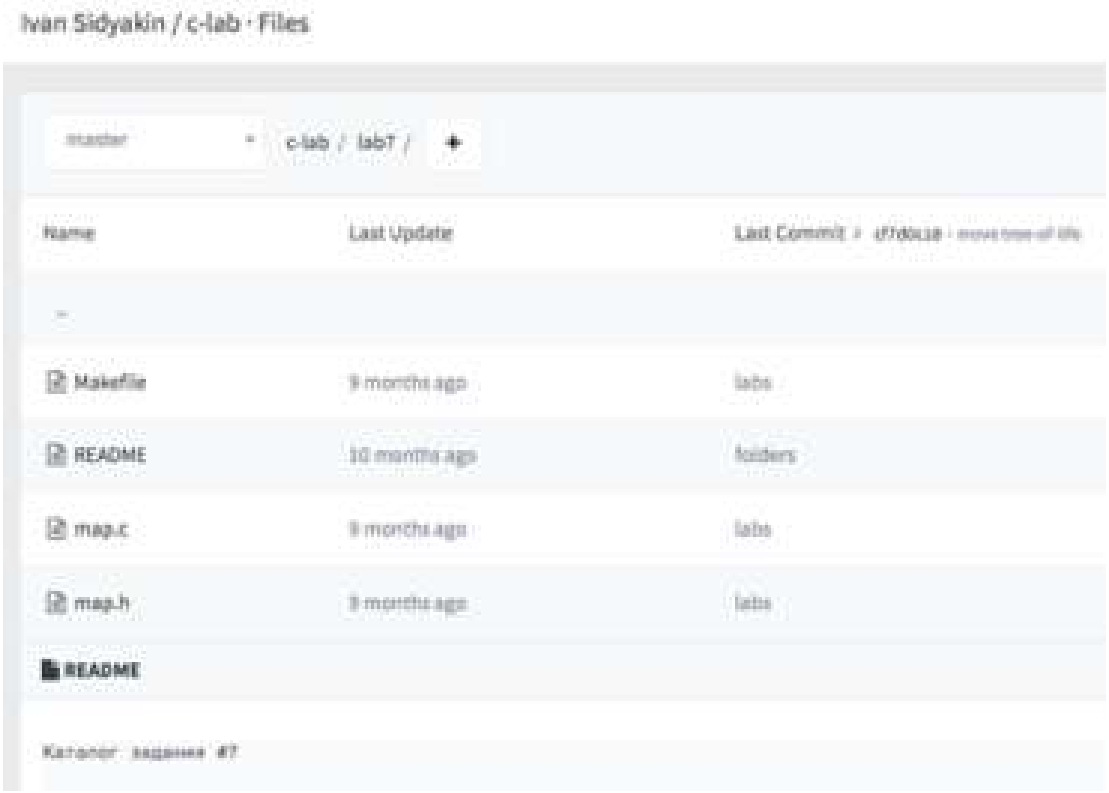

Рисунок 2. Файлы необходимые для выполнения одного из заданий курса

Полученный описанным выше способом проект используется как шаблон для создания персональных проектов учащихся. Каждый учащийся получает свою собственную копию проекта шаблона, в которой он может редактировать и добавлять файлы задания. Такая копия проекта в Git называется fork. Обычно fork создается для каждого члена команды работающей над проектом. Fork это самостоятельный проект, доступ к которому, как правило, имеет сам разработчик и кроме него (не обязательно), только администратор основного проекта. Разработчик вносит изменения в этот проект и, когда поставленная перед ним задача решена, синхронизирует изменения в своей копии с основным проектом. Передать эти, сделанные в fork изменения, в основной проект можно, как правило, только с разрешения администратора проекта. Все это делается для того, чтобы всегда, при внесении изменений в разрабатываемый продукт, основной проект поддерживался в рабочем состоянии. Этом свойством fork мы воспользуемся для клонирования проектов учебных заданий каждому учащемуся. Загрузка изменений сделанных учащимися, при выполнении своих заданий, из fork назад, в основной проект, конечно не требуется и, более того, это свойство необходимо отключить. В GitLab, для этого, достаточно, чтобы администратор системы установил для учащихся права доступа к их fork проектам, запрещающие им обратную синхронизацию с основным проектом.

Для проверки заданий, преподаватель должен загрузить fork проекты всех учащихся на свой компьютер. Используя всего несколько команд, каждый учащийся может записать результаты своей работы в репозиторий собственного fork проекта, а преподаватель, так же с помощью нескольких команд git, может загрузить текущую версию заданий всех учеников на свой компьютер для проверки. Аккаунт преподавателя должен быть добавлен в группу пользователей каждого из fork проектов. Иначе говоря, у каждого fork проекта в GitLab должно быть два разработчика: один из учащихся и преподаватель, тогда они оба получают доступ к репозиторию проекта.

Веб-приложения GitLab, кроме пользовательского интерфейса для управления системой “вручную”, имеет программный интерфейс Web-API, который можно использовать для автоматизации некоторых нужных нам действий. В частности добавление пользователей, создание и настройку их fork проектов, а также загрузку выполненных заданий на компьютер преподавателя можно запрограммировать. Приведем, с необходимыми пояснениями, несколько полезных скриптов, которые можно запускать из командной строки ОС Linux. Скрипт использует программу curl [10] для вызова функций Web-API GitLab.

Рассмотрим вначале скрипт, использующийся для регистрации в системе нового учащегося и создания для него fork проекта. Через аргументы командной строки этот скрипт получает адрес электронной почты нового пользователя, логин (USER_NAME) и имя.

```
#!/bin/bash
USER_EMAIL=$2
USER_NAME=$3
NAME=$1
```

GitLab API содержит функции, которые позволяют просмотреть список пользователей и проектов. С помощью этих функций надо заранее выяснить индекс проекта шаблона (ORIGIN) и индекс пользователя который будет преподавателем курса. Индексы выводятся вместе с другой информацией о проектах и пользователях. В этом примере индекс курса 14. Это проект, структура которого приведена выше на рисунках 1 и 2. Индекс аккаунта преподавателя равен 2. TOKEN это строка использующаяся для авторизации в GitLab при обращении к Web-API. Уникальный токен генерируется для каждого пользователя. Здесь надо указать токен администратора GitLab, имеющего право добавлять пользователей и создавать новые проекты. Для того чтобы добавить нового пользователя, требуется указать его пароль. В скрипте можно задать какой нибудь неизвестный никому кроме вас и сложный пароль по умолчанию. Пользователи, впоследствии, могут получить доступ к GitLab воспользовавшись функцией восстановления пароля. Таким образом указанный здесь пароль по умолчанию временный и не должен нигде использоваться. В переменной URL сохраним адрес сервера GitLab для того, чтобы подставлять его ниже в коде скрипта, там где это потребуется.

```
ORIGIN=14
TEACHER=2
TOKEN=<токен>
PASSWORD=<пароль нового пользователя по умолчанию>
URL= <URL сервера GitLab>
```

Первое обращение к Web API добавляет нового пользователя. Заметим, что TOKEN администратора передается в заголовке HTTP запроса. В теле запроса передается адрес электронной почты логин и имя (ФИО) нового пользователя.

```
echo ADD USER
user=`curl -sS --header "PRIVATE-TOKEN: $TOKEN" --data
"email=$USER_EMAIL&password=$PASSWORD&username=$USER_NAME&name
=#$NAME&projects_limit=1&can_create_group=false"
"$URL/api/v3/users"`
uid=`echo ${user} | jq -r '.id'`
if [ "$uid" == "null" ]; then
echo $user
echo $uid
echo $USER_NAME
echo "Failed to create user"
#exit
fi
```

Если новый пользователь успешно создан, переходим к созданию fork проекта. Для этого требуется авторизоваться в GitLab от имени нового пользователя с помощью Web API session. Из ответа извлекаем token пользователя и его индекс, которые нам понадобятся позднее. Собственно вызов session нужен только для того, чтобы получить эту информацию.

```
    echo LOGIN
    login=`curl -sS --data
"id=$uid&login=$USER_NAME&password=$PASSWORD"
"https://hub.iu3.bmstu.ru:2443/api/v3/session"`
    pt=`echo ${login} | jq -r '.private_token'`
    uid=`echo ${login} | jq -r '.id'`
    if [ "$pt" == "null" ]; then
    echo pt = $pt
    echo login = $login
    echo user name = $USER_NAME
    echo "Failed to login"
    exit
    fi
```

Fork проекта надо делать от имени пользователя, для которого делается этот fork проект. Поэтому здесь указываем его token. Проект шаблон с индексом ORIGIN должен быть настроен так, чтобы его fork мог сделать любой пользователь зарегистрированный в системе. Это надо указать в настройках проекта шаблона при его создании. Из ответа извлекаем индекс созданного fork проекта. Он понадобится для изменения настроек.

```
    echo FORK PROJECT
    proj=`curl -sS --header "PRIVATE-TOKEN: $pt" --data
"id=$ORIGIN" "$URL/api/v3/projects/fork/$ORIGIN"`
    pid=`echo ${proj} | jq -r '.id'`
    if [ "$pid" == "null" ]; then
    echo pt = $pid
    echo $proj
    echo $USER_NAME
    echo "Failed to fork project"
    exit
    fi
```

Fork проекта шаблона для нового пользователя создан. Меняем две настройки по умолчанию. Скрываем проект от всех кроме его участников и запрещаем запросы на синхронизацию с основным проектом (merge request).

```
    echo EDIT PROJECT
    eproj=`curl -sS -X PUT --header "PRIVATE-TOKEN: $TOKEN" --
data
"id=$pid&visibility_level=0&issues_enabled=false&merge_request
s_enabled=false" "$URL/api/v3/projects/$pid"`
    epid=`echo ${eproj} | jq -r '.id'`
    if [ "$epid" == "null" ]; then
    echo $eproj
    echo $USER_NAME
    echo "Failed to set fork settings"
    exit
    fi
```

По умолчанию, пользователь, для которого создается fork проект, получает над этим проектом полный контроль. Мы же хотим предоставить максимальные права для преподавателя и ограничить права учащегося. Например, учащийся не должен менять настройки проекта, которые были заданы на предыдущем шаге. Для этого права нового пользователя в проекте снижаются до уровня “Developer”.

```
    echo REDUCE USER ACCESS RIGHTS TO DEVELOPER
    dv=`curl -sS -X PUT --header "PRIVATE-TOKEN: $TOKEN" --
data "id=$pid&user_id=$uid&access_level=30"
"$URL/api/v3/projects/$pid/members/$uid"`
    dev=`echo ${dv} | jq -r '.id'`
    if [ "$dev" == "null" ]; then
    echo $dv
    echo $USER_NAME
    echo "Failed to set user rights"
    exit
    fi
```

Последний шаг это добавление к проекту преподавателя с максимальным набором прав.

```
    echo ADD TEACHER
    team=`curl -sS --header "PRIVATE-TOKEN: $TOKEN" --data
"id=$pid&user_id=$TEACHER&access_level=40"
"$URL/api/v3/projects/$pid/members"`
    tid=`echo ${team} | jq -r '.id'`
    if [ "$tid" == "null" ]; then
    echo $team
    echo $USER_NAME
    echo "Failed to add teacher"
    exit
    fi
    echo SUCCESS $USER_EMAL
```

Для загрузки fork проекта учащегося на компьютер преподавателя, достаточно выполнить команду

```
    git clone git@<url>:<веб ссылка на проект>.git <имя поль-
зователя>
```

Например, так:

```
    git clone git@mygitlab.bmstu.ru:user1/c-lab.git user1
```

Эта команда копирует проект из репозитория в файловую систему компьютера. Можно написать скрипт, который каждый раз заново копирует проекты всех учащихся, предварительно стирая старые версии проектов.

Крайне важно организовать учебную работу так, чтобы преподаватель и учащиеся использовали одни и те же инструменты для проверки и выполнения заданий. В этом случае снижается вероятность выставления некорректных оценок. Преподаватель и учащийся видят один и тот же результат исполнения заданий на свих компьютерах, используют одинаковую кодировку для записи исходного кода программ. Эту задачу рекомендуется решать с помощью технологий виртуализации [5]. Для выполнения практических заданий курса заранее создается виртуальная машина, со всем необходимым набором инструментов, включая, например, git,текстовый редактор, компилятор, отладчик, средства составления документации [6], средства тестирования [7]. Учащиеся используют свои копии виртуальной машины для выполнения заданий и копирования результатов работы в репозиторий GitLab. Преподаватель использует свою копию виртуальной машины для копирования из репозитория и проверки работ учащихся. Дополнительно, средства Gitlab позволяют преподавателю публиковать учебные материалы на Wiki страницах и комментировать задания, с помощью встроенной системы отслеживания ошибок.

Список литературы

1. Документация и дистрибутивы GitLab. Режим доступа: http://www.gitlab.com (дата обращения 24.09.2016).
2. Документация Git. Режим доступа: https://git-scm.com/book/ru/v1/Введение-Основы-Git (дата обращения 24.09.2016).
3. Система поддержки обучения Moodle. Режим доступа: https://moodle.org (дата обращения 24.09.2016).
4. ОС LInux Ubuntu. Режим доступа: http://www.ubuntu.com/ (дата обращения 24.09.2016).
5. Oracle Virtual Box. Режим доступа: https://www.virtualbox.org/ (дата обращения 24.09.2016).
6. Doxygen. Система документирования исходного кода программ. Режим доступа: http://www.stack.nl/~dimitri/doxygen/ (дата обращения 24.09.2016).
7. Valgrind. Обнаружение утечек памяти в программах. Режим доступа: http://valgrind.org/ (дата обращения 24.09.2016).
8. Википедия. Системы отслеживания ошибок (bug trackers). Режим доступа: https://ru.wikipedia.org/wiki/Система_отслеживания_ошибок (дата обращения 24.09.2016).
9. Bash Reference Manual. Режим доступа: https://www.gnu.org/software/bash/manual/bash.html (дата обращения 24.09.2016).
10. CURL. Программа для передачи данных адресуемых URL. Режим доступа: https://curl.haxx.se/ (дата обращения 24.09.2016).

e l e c t r o n i c j o u r n a l
SCIENCE & EDUCATION
Bauman Moscow State Technical University. El № FS 77 - 48211. ISSN 1994-0408

Using GitLab Source Control Software in Teaching Programming

10, October 2016
DOI: 10.7463/1016.0848154

I.M. Sidyakin[1,*]

[1]Bauman Moscow State Technical University,
Moscow, Russia
[*]ivan.sidyakin@gmail.com

The article deals with using a GitLab control system to complete and test the practical assignments in programming. A brief description of the system features that can be used in the teaching process, including the exchange of source assignment codes between teacher and students, publication of learning materials and messaging.

A developed technique for using this system to manage the remote laboratory works is presented. The article considers a practical implementation of the course of study based on the system GitLab, including the GitLab server installation and setting, development of the study project template, method to create study projects from this template, as well as to synchronize the source code on the PCs of students and faculty staff.

The article explores a subject of the student's project control automation. Gives the explained examples of programs in the Linux commandshell scripting language that implement the student's logon, creation and settings of the study projects using the Web API GitLab software interface. Developed scripts are intended for a teacher or administrator of the system and enable us to simplify a setting procedure of the study course resources in programming and a test procedure of assignments.

In conclusion, the article makes recommendations for management of education activity using virtualization technologies, describes a flow process of completing and transferring assignments for test through a VirtualBox virtual machine and a GitLab server. Provides the links to the recommended aids to manage the educational process and design tools that can be used in teaching programming.

References

1. *Dokumentatsiia i distributivy GitLab* [Documentationa and distributions GitLab]. Available at: http://www.gitlab.com, accessed 24.09.2016.

2. *Dokumentatsiia Git* [Documentation Git]. Available at: https://git-scm.com/book/ru/v1/ , accessed 24.09.2016.
3. *Sistema podderzhki obucheniia Moodle* [The study support system Moodle]. Available at: https://moodle.org, accessed 24.09.2016.
4. OC LInux Ubuntu. Available at: http://www.ubuntu.com/, accessed 24.09.2016.
5. Oracle Virtual Box. Available at: https://www.virtualbox.org/ , accessed 24.09.2016.
6. *Doxygen*. *Sistema dokumentirovaniia iskhodnogo koda programm* [Doxygen. System of documentation of program source code]. Available at: http://www.stack.nl/~dimitri/doxygen/, accessed 24.09.2016.
7. *Valgrind. Obnaruzhenie utechek pamiati v programmakh* [Detecting memory leaks in programs]. Available at: http://valgrind.org/, accessed 24.09.2016.
8. *Vikipediia. Sistemy otslezhivaniia oshibok* [Vikipedia. System bug tracking]. Available at: https://ru.wikipedia.org/wiki/, accessed 24.09.2016.
9. Bash Reference Manual. Available at: https://www.gnu.org/software/bash/manual/bash.html
10. *CURL. Programma dlia peredachi dannykh adresuemykh URL* [Pprogram to transfer data to addressable URL]. Available at: https://curl.haxx.se/ , accessed 24.09.2016.