Для запуска сети необходима стабильная работа уровня L2 коммутации, в которой задействованы устройства SW1, SW2, SW3. Задействованы технологии Etherchannel via LACP, VTP, Rapid PVST+. Поэтому в качестве первого шага стоит настроить сегмент коммутации.

После установления соединения можно переходить к настройке IP-стека и протоколов маршрутизации.

Распределение сабинтерфейсов по VLAN в смежных сетях позволяет организовать работу L3-схемы. Создаются туннельные соединения.

Последним шагом реализации решения рекомендуется проводить интеграцию защиты каналов связи и доменов маршрутизации.

Реализация VoIP сервисов и других сервисов прикладного уровня можно осуществлять после запуска IP- стека на участках сети, наиболее приближенных к их контроллерам.

А.Ю. Ковалёв (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **В.В. Грищенко**, ст. преподаватель

ПРАКТИКА ПРИМЕНЕНИЯ МЕТОДОЛОГИИ DEVOPS НА ПРОЕКТАХ В СТАДИИ РАЗРАБОТКИ И ПОДДЕРЖКИ

Для сокращения времени требуемого на выполнение проекта, необходимо применять практики методологии DevOps. Специалисты DevOps предлагают команде разработчиков контролируемую среду разработки и отладки конечного продукта с помощью специализированного ПО: виртуализации, контейнеризации, контроля версий, системы непрерывной интеграции и развертывания продукта. Процесс непрерывной интеграции и развертывания состоит из комплекса процессов CI/CD: continuous integration, continuous delivery и continuous deployment (рисунок 1).

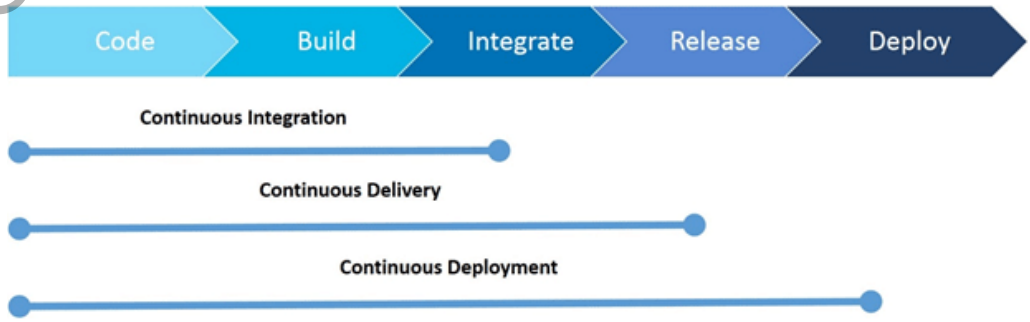

Рисунок 1 – Цикл разработки программного обеспечения

В компании INTERVALE в рамках работы над проектами «Технологии обеспечения безопасных платежей» используется следующее программное обеспечение: система непрерывной интеграции TeamCity, система управления конфигурациями Ansible, система виртуализации VMware, среда разработки IntelliJ IDEA.

В рамках работы над продуктом возникает необходимость производить своевременное обновление среды разработки и внедрения. Существует 2 вида обновления: ручное – производится непосредственно специалистом внедрения и автоматическое – осуществляется с помощью скриптов. С целью сокращения времени простоя разрабатываемого продукта применяется автоматическое обновление, которое позволяет сократить время, потраченное на развертывание обновленной сборки, что в свою очередь ускоряет процесс выполнения проекта

Рассмотрим пример организации автоматического обновления среды разработки в рамках системы непрерывной интеграции с помощью системы управления конфигурациями Ansible (листинг 1).

Ansible — система управления конфигурациями, написанная на Python. Используется для автоматизации настройки и развертывания программного обеспечения.

Листинг 1 – Anisble-скрипт автоматизации обновления сервиса платформы электронной коммерции

```
---
- name: getting build number from url teamcity
uri:
url:
"{{tc_url}}/httpAuth/app/rest/buildTypes/id:{{build_type_id}}/
builds/number:{{build_num}},branch:default:any/id"
user: "{{tc_user}}"
password: "{{tc_pass}}"
return_content: yes
validate_certs: no
register: id_build

- name: download zip-archive on back
get_url:
url:
"{{tc_url}}/repository/download/{{build_type_id}}/{{id_build.c
ontent}}:id/application/app-{{version}}.zip"
url_username: "{{tc_user}}"
url_password: "{{tc_pass}}"
validate_certs: no
dest: "{{dest_folder_back}}/distrib/"

- name: stoping application.
Service:
```

```
name: "back"
state: stopped

- name: create backup.
Shell: mv "{{dest_folder_back}}/back/"
"{{dest_folder_back}}/#{{old_build_num}}_back/"
ignore_errors: yes

...
```

Представленный участок скрипта написан с применением декларативного языка разметки для описания конфигурационного файла. В данном скрипте описываются этапы выполнения автоматического обновления системы. Каждый этап является самостоятельным набором команд, которые выполняются один за другим. При выполнении данного файла, на экран пользователя выводится информация о ходе выполнения обновления (рисунок 2).

```
PLAY [Upload files      on test stand    (back)] ***********

TASK [Gathering Facts] ********************************************
ok: [     _back]

TASK [     _back : getting bild number from url teamcity] ***
ok: [     _back]

TASK [     _back : download zip-archive on back] ************
changed: [     _back]

TASK [     _back : stoping ecp.] ****************************
changed: [     _back]

TASK [     _back : create backup.] **************************
changed: [     _back]

TASK [     _back : unarchive file with app] *****************
changed: [     _back]
```

Рисунок 2 – Ход выполнения этапов обновления

После выполнения всех этапов обновления система выводит на экран пользователя краткую информацию о выполненном обновлении, а именно: сколько было изменений, сколько элементов не было изменено и количество ошибок.

Итогом данной работы является сокращение времени затрачиваемого на обновление. В сравнения с ручным обновлением, которое занимало от 15 минут DevOps-инженера, автоматическое обновление, позволяет производить все необходимые действия, направленные на развертывание обновленной конфигурации сервиса, в течение 3-5 минут.

Описанный механизм и приведенные участки кода прошли тестирование и используются в рамках работы над проектами: «Технологии реализации безопасных платежей».

А.Д. Ковальчук (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **М.И. Жадан**, канд. физ.-мат. наук, доцент

РАЗРАБОТКА WEB-ПРИЛОЖЕНИЯ ДЛЯ ХОСТИНГА ИЗОБРАЖЕНИЙ

Хостинг изображений – это ресурс, который позволяет загружать графические файлы на удалённый сервер. Пользователь может зайти на сайт, разместить изображения, настроить уровень доступа к ним и получить ссылку на загруженный материал.

При первом открытии хостинга пользователь попадает на главную страницу ресурса (рисунок 1). Далее можно произвести регистрацию или авторизацию. Авторизованный пользователь может настроить аккаунт или выйти из системы. Загружать изображения можно и без авторизации, но в таком случае нельзя определять уровень доступа к файлам. Также авторизованный пользователь может просмотреть все свои файлы, т.е. ему не нужны отдельные ссылки для доступа к каждому изображению.

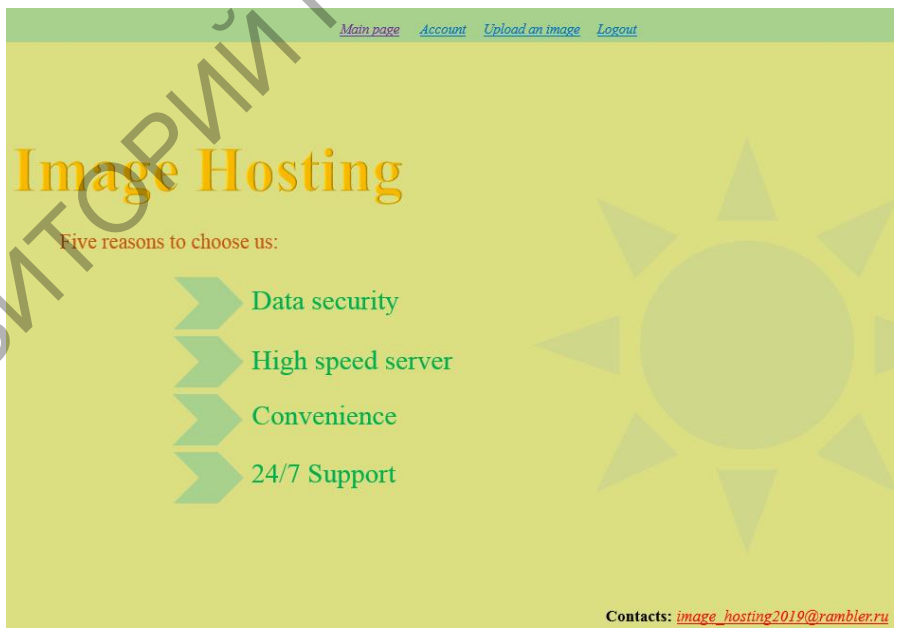

Рисунок 1 – Главная страница хостинга

Разработанное веб-приложение состоит из небольшой серверной части, написанной на Java и массивной клиентской, с которой конеч-