SAVONIA

University of Applied Sciences

THESIS – BACHELOR'S DEGREE PROGRAMME
TECHNOLOGY, COMMUNICATION AND TRANSPORT

GITHUB ENTERPRISE AND MIGRATION OF CI/CD PIPELINES FROM AZURE DEVOPS TO GITHUB

AUTHOR Olli Rautiainen

Field of Study Technology, Communication and Transport	
Degree Programme Degree Programme in Information Technology	
Author Olli Rautiainen	
Title of Thesis GitHub Enterprise and Migration of CI/CD Pipelines from Azure DevOps to GitHub	
Date 3 May 2023	Pages/Number of appendices 37/0
Client Organisation Ponsse Plc	

Abstract

DevOps platforms help to enable automation to different processes in software development lifecycle. GitHub Enterprise and Azure DevOps are both Microsoft owned DevOps platforms. They provide similar features to business customers, but the implementation of CI/CD features varies. Artificial intelligence products like GitHub Copilot extend automation to developer workflows and help to code faster and focus on larger problems rather than smaller repetitive tasks.

The purpose of this thesis was to research GitHub Enterprise platform, the CI/CD capabilities of GitHub Actions and the privacy state of GitHub Copilot for the client organization. The goal was to find out if these platforms and features are suitable for the client and to estimate the workload that must be done to migrate the client's current CI/CD pipelines from Azure DevOps to GitHub Actions. The thesis was commisioned by Ponsse Plc.

The work was conducted between January and April 2023. First, the platforms were studied from the official documentation and from similar migration processes and guides. Next, GitHub Enterprise Server was deployed and GitHub Actions CI/CD features were tested and compared to Azure DevOps. The automated migration process was tested with GitHub Actions Importer. Lastly, GitHub Copilot privacy statements were reviewed.

In conclusion, GitHub Enterprise platform had the required capabilities and features. GitHub Actions had very similar CI/CD features than Azure DevOps and was equally capable of doing the required tasks. This work demonstrated some differences in the files that define the pipelines and conversions between platforms. GitHub Actions Importer converted parts of the pipelines automatically and helped to plan the overall migration process. There still remained a lot of manual work in the process, especially when converting the Azure DevOps custom tasks. Therefore, Actions Importer needs more testing with the actual pipelines of the client to get a more precise estimate of the required manual workload. GitHub Copilot for Business version had the strictest privacy statements. GitHub stated that it does not collect, share or use the code of the Business user. The code is processed in real time and is not retained for further use. GitHub collects user engagement data from all Copilot users. This data is used for product improvements and research purposes and is shared with Microsoft and OpenAI.

Keywords
DevOps, automation, continuous integration, continuous delivery, artificial intelligence

CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

DevOps platforms are a core part of the business's software development lifecycle and play a great role in how successful it can be. Businesses want to deploy new features and products more quickly and as efficiently as possible. They implement automation and orchestration tools to keep up with this rapid and evolving cycle and at the same time try to manage the whole process and recover quickly from errors.

69 per cent of the 33 000 respondents of the Accelerate State of DevOps survey stated that they deploy their primary product to production or release it to end users between once per week and once per month. 11 per cent of the respondents can deploy on-demand with a failure rate of 0 to 15 per cent. (DevOps Research and Assessment, 2022.)

Humans are prone to errors, so tools that enable developer teams to easily implement automation to their processes mitigate the risks substantially. Continuous integration and continuous delivery (CI/CD) pipelines consist of a series of automated steps for developing, testing and delivering software. Machine learning (ML) and artificial intelligence (AI) help to automate repetitive tasks in coding and free the developers to concentrate on more difficult problems and innovation.

The main point of this thesis is to research GitHub Enterprise platform suitability for Ponsse Plc's digital services. Suitability is mainly examined based on the CI/CD features of GitHub Actions. The second goal is to estimate the amount of work that must be done to migrate current CI/CD pipelines from Azure DevOps to GitHub Actions and test the available tools for migration process. This thesis also reviewed GitHub Copilot services current privacy state regarding business customers. The thesis was commissioned by Ponsse Plc. The author has worked in Ponsse's DevOps CICD team part-time alongside studies for the last year.

The research questions are:

- Does GitHub Enterprise Server platform meet the Ponsse DevOps requirements?
- Does GitHub Actions CI/CD workflows meet the requirements of Ponsse software development?
- Estimate the workload of migrating CI/CD pipelines from Azure DevOps to GitHub Actions?
- What is the current privacy state of GitHub Copilot?

This thesis concentrates mostly on CI/CD features of the GitHub Enterprise and many features were left out of scope of this work or just mentioned briefly. In many cases the only up-to-date source was the service provider's official documentation, so material was gathered mostly from GitHub and Microsoft documentation. To speed up the process and limit the configuration work, all testing of the services and tools was made in the author's personal environments and subscriptions in Azure, Azure DevOps and in GitHub Enterprise outside of Ponsse's networks and systems.

2 BACKGROUND

Microsoft is the owner of both developer platforms Azure DevOps and GitHub. It purchased GitHub in 2018 and from that time the user amount has grown from 31 million developers to 100 million. Previously GitHub was mostly a code repository. Now GitHub has developed tools and features like CI/CD platform, project management boards and security tools. GitHub Enterprise version serves the requirements for even the largest companies like Ford and Spotify. (Dohmke, 2023; GitHub Inc, s.a.)

Azure DevOps is Microsoft's own developer platform with very similar capabilities. They directly compete for the same customers but still both offer tools for platforms to work together and complement each other. Microsoft has kept GitHub as an independent company for now.

GitHub CEO Thomas Dohmke stated in 2022 that,

"We kept GitHub GitHub and it remains this independent entity within Microsoft similar to LinkedIn. -- You don't see more Microsoft in GitHub.com than you saw four years ago and that has helped us to continue to grow and we're very excited where this is going." (Lardinois, 2022.)

Microsoft is large owner of OpenAI, a non-profit company that develops artificial intelligence services such as GitHub Copilot, DALL-E 2 and ChatGPT. Microsoft announced in March 2023 its own AI-powered services Microsoft 365 Copilot and Microsoft Dynamics 365 Copilot. When using AI services like GitHub Copilot, developers are more productive, more fulfilled in their job and more efficient. (Kalliamvakou, 2022; Stallbaumer, 2023; OpenAI Corporation, s.a.) Remains to be seen in what direction Microsoft steers these developer platforms and AI products in the future.

2.1 Ponsse Plc

Ponsse Plc is a Finnish cut-to-length forest machine manufacturer. The company was founded in 1970 by Einari Vidgren and is based in Vieremä, Finland. Ponsse Plc is the parent company of Ponsse Group and the subsidiaries are Ponsse AB, Sweden; Ponsse AS, Norway; Ponssé S.A.S. France; Ponsse UK Ltd., Great Britain, Ponsse Czech s.r.o., Czech Republic; Ponsse North America, Inc., the United States of America; Ponsse Latin America Ltda, Brazil; OOO Ponsse Russia; Ponsse Asia Pacific Ltd, Hongkong; Ponsse China Ltd, China; Ponsse Uruguay S.A., Uruguay; Ponsse Chile SpA, Chile and Epec Oy, Finland. In the year 2022 net sales amounted to EUR 755.1 million and net result was EUR 34.2 million. The average staff during the year was 2016 people. (Ponsse Plc, 2023.)

One of Ponsse's core values is innovation with continuous improvement of products, services and processes (Ponsse Plc, s.a.). Ponsse currently uses Azure DevOps services and wants to actively keep up with the latest changes and capabilities of Microsoft products to determine what are the best tools for their business.

3 GITHUB ENTERPRISE

GitHub (also GitHub.com) is a developer platform that supports the entire software development lifecycle of individual and business customers. Included products are version control, project management and planning, pipelines for automated builds, tests and deployments and package management for hosting software. GitHub uses Git as a version control system. By purchasing GitHub Enterprise license, these features can be hosted either in cloud or in self-hosted server and have more managed solution based on business requirements. More advanced features related to code security and premium support can optionally be added. (GitHub Inc, s.a.)

GitHub offers Software as a service (SaaS) model Enterprise Cloud, that extends GitHub.com with features that suit most of the large businesses. Features include more managed administration of the organizations under the enterprise account, improved visibility of the organizations, private and public repositories, enterprise grade authentication and user management. (GitHub Inc, s.a.)

GitHub Enterprise Server (GHES) is a self-hosted platform which can be deployed either as an on-premises solution or to supported cloud environments. Because the GHES instance is deployed to the company's own infrastructure, access and security policies can be managed more thoroughly. GHES is most suitable for companies that must be compliant with regulations, laws or guidelines created by government legislations or other regulatory bodies. Geographical location can also be chosen to specify where the code is located. GHES can connect to GitHub Enterprise Cloud and GitHub.com with optional GitHub Connect service and enable additional features such as easier use of public GitHub.com actions. (GitHub Inc, s.a.)

The third option for enterprises is GitHub AE. It is GitHub managed cloud service which has strict security and compliance requirements (GitHub Inc, s.a.). Currently it is in limited release so this thesis will not examine it further.

Ponsse requires its product development data to be stored inside Europe. This is why next topics are reviewed based on GitHub Enterprise Server capabilities and functionalities.

3.1 Enterprise Server distribution and upgrades

GHES is distributed as a self-contained virtual appliance. The operating system is a modified Debian 10 (Buster) Linux installed with only the necessary applications and services. GitHub does not support modifying or upgrading of the system, so it provides a regular cycle of releases and patch updates and needed security patches outside the normal cycle. Enterprise administrators can decide if and when they want to upgrade or patch the instance. Feature updates usually cause a few hours interruption to service. Patch releases interruption to service is generally a few minutes of downtime. If the instance version is more than two releases behind the current feature release, the Upgrade Assistant must be used. It shows the correct upgrade order of release versions. (GitHub Inc, s.a.)

3.2 Enterprise account

Before the enterprise can install GHES instance it must purchase GitHub Enterprise license and create an enterprise account. Account can be created for GitHub Enterprise Cloud or Server or to be synced to use both. Administrators of the enterprise account can then manage all the organizations that the enterprise owns. If the account is only for GHES, administrators can view and manage their own enterprise and organization memberships, license usage, authentication, certificates and general policies from the GHES instance. If the account is for GitHub Enterprise Cloud or synced to it these can also be managed from github.com. (GitHub Inc, s.a.)

3.3 Roles and teams

Roles and teams are key elements when controlling access to enterprises' data. Every user that has been granted access to the organizations owned by the enterprise is a member of the enterprise. Roles and teams are a way to control what members can do in the scopes of enterprise, organization, or repository. (GitHub Inc, s.a.) Next is a list of different roles that members can be assigned to.

Enterprise owner:

- Oversee and manage the enterprise.
- Manage and assign enterprise administrators.
- Can join any organization and manage its settings.
- Enforce policies to all organizations.

Enterprise member:

- Cannot manage enterprise settings.
- Can be assigned to be an organization owner.
- Can be assigned as a member of an organization.
- Can have different levels of access to repositories and resources.

Organization owner:

- Create teams for the organization.
- Manages any repository inside the organization.
- Manages the roles of any member or owner of the organization.
- The owner's role should be assigned to a minimum of two people.

Organization member:

- Can have individually assigned permissions.
- Can have permissions assigned through teams.
- Can be assigned to be a member of teams.
- Has by default permission to create repositories and project boards.

A team is a group of members in an organization. It is a way to manage access and permissions to repositories, handle discussions and collaboration and organize people in a way that it reflects the

company's structure. Teams can be either visible to all organization members or secret and only visible to the members of the team. All the visible teams can be nested to parent and multiple child teams, for example *Employees > R&D > Project1 > Developers > Testing.* (GitHub Inc, s.a.)

GitHub Enterprise Server has a different organization structure than Azure DevOps. It might not be the best approach to just directly convert Organizations and repositories in Azure DevOps to GitHub organizations and repositories. Azure DevOps has a concept of projects in its structure, but GitHub does not. The use of teams is a way to control repository access in cases where there are just one or a few organizations. Teams work similar way as Azure DevOps projects. It is up to the enterprise to decide how they want to structure their GitHub organizations. GitHub recommends that the number of organizations under the enterprise account in GitHub should be as little as possible. Fewer organizations enable better collaboration between teams, finding resources is more efficient and communication is easier. (GitHub Inc, s.a.)

3.4 Deployment environments

GHES can be deployed to virtualization hypervisors in on-premises datacenters or to public cloud services. Supported virtualization hypervisors are Microsoft Hyper-V, OpenStack KVM and VMware ESXi. Supported cloud services are Amazon Web Services (AWS), Google Cloud Platform (GCP) and Microsoft Azure. (GitHub Inc, s.a.)

There are multiple factors to consider when deciding which environment to choose:

- Previous experience and usage.
- Minimum hardware requirements.
- Networking, load balancer and administrative access methods.
- External storage requirements for GitHub Actions.
- Availability and backups.
- Security requirements.
- External services.

3.5 Monitoring and fault tolerance

GHES has a built-in monitoring dashboard that administrators can access from the instances website. It shows instance system health, CPU and storage usage and response times of the application and authentication. System logs of the instance are retained for seven days and can be forwarded to other systems or servers if longer retention is needed. Audit logs can be streamed nearly in real time to different streaming endpoints in Azure, Amazon, Google, Datadog and Splunk service providers. Audit log streaming includes data about access settings, permissions changes, added or removed users, API requests and Git events. (Boyle et al. 2023; GitHub Inc, s.a.)

GitHub offers tools to manage and design the availability and fault tolerance of the service. Even though GHES is a standalone instance, enterprise can increase fault tolerance and availability by implementing passive replicas of the instance. Passive replicas mitigate the impact of system, network and virtualization host system failures. Replica appliances mirror the primary system but run in a

replication mode and only on limited services. Failure time is dependent on the manual work of switching instance statuses. In case of failures, replicas must be manually promoted to primary status and traffic must be redirected. (GitHub Inc, s.a.)

Performance can be increased by scaling the service with clustering. GHES is a set of services normally running in a single node. In the cluster setup, the load balancer automatically distributes the load across multiple nodes which each is serving the desired services. Instances of the same service are mostly equal with two exceptions being MySQL and Redis servers. These run on a single primary node and data is copied to one or more replica nodes. (GitHub Inc, s.a.)

Clustering does not replace high availability and in most business cases it is simpler to just have primary and passive replicas of the instance. Clustering requires a lot of planning in cases of management, installation, disaster recovery and upgrades and in many cases, it adds too much complexity to the setup to be beneficial.

3.6 Runners

Runners are computers or virtual machines that contain the runner application and required tools, packages and settings to run the workflows they are assigned to. Runners can be either GitHub-hosted or self-hosted. GitHub-hosted runners come with pre-installed tools either in Ubuntu Linux, Windows or macOS operating systems. At the time of writing the thesis, GitHub Enterprise Server cannot use GitHub-hosted runners, but the feature is on GitHub roadmap and is expected to be enabled in the future (GitHub Inc, 2020)

GitHub Enterprise Server can use self-hosted runners. These are computers or virtual machines deployed, managed and maintained by the enterprise admins (Figure 1). Runners can be assigned to enterprise-level so that the organizations and repositories can access them. This makes the management of the runners more centralized. Or the runner can be accessible only to a specific repository. Runners can also be divided into groups for access control and labeled based on their capabilities. Deployment can be done to local machines or cloud services and can be fully customized to meet hardware, software, operating system and security requirements. Self-hosted runners do not require public internet access and they can directly communicate with just the GHES instance. Automatic access to github.com actions can be enabled and configured if needed.

Figure 1 Self-hosted runner configuration experience in Ubuntu virtual machine (Rautiainen, 2023)

3.7 GitHub Connect

GitHub Connect enables some GitHub.com features in GHES. This does not open the instance to the public internet, expose private data to GitHub.com users or transmit repositories, issues or pull requests to GitHub.com. All the features can be enabled individually.

Features include:

- Automatic user license sync between GHE Server and GHE Cloud.
- Dependabot feature to find and fix dependency vulnerabilities.
- Allow access to GitHub.com actions.
- Server statistics to aggregate and analyze GHES data.
- Unified search that allows users to search also from GitHub.com repositories.
- Unified contributions that enable contribution graphs on GitHub.com.

3.8 Dependabot alerts

Dependabot alerts is an additional feature of the GitHub Enterprise Server and can be enabled by the enterprise administrators. It detects the insecure and vulnerable dependencies in the code against the GitHub Advisory Database. Advisory must have been already reviewed by GitHub to cause an alert. Alerts are displayed in the security tab and in the dependency graph of the repository. (GitHub Inc, s.a.)

Dependabot alerts add visibility to possible vulnerable interactions and hidden assets in the code. Log4Shell vulnerability in late 2021 was one of the largest supply chain vulnerabilities so far. It un-

derlined that security cannot just rely on security teams but must be part of the development lifecycle. As stated in the article of the lessons learned by the vulnerability by Daniel Thomas "What you can't see, you can't fix". (Thomas, 2022.)

Finnish National Cyber Security Centre stated in 2021 that it is vital in organizations to have clear policies of the third-party libraries usage and update procedures. And also, to know how its own systems are built and what libraries they are dependent on. (Finnish Transport and Communications Agency, National Cyber Security Centre, Traficom, 2021.)

3.9 Rate of releases and feature updates

The release cycle for GitHub Enterprise is far shorter than Azure DevOps has. From January 2022 to March 2023 Azure DevOps published 20 releases while at the same time GitHub published 108 releases for Enterprise. These releases include feature updates, bug fixes and security vulnerabilities to both cloud and on-premises services of the platforms.

3.10 Ponsse requirements

GitHub Enterprise Server meets the requirements of Ponsse software development in case of features in scope of this thesis. Server can be deployed in on-premises datacenters and in all major cloud providers datacenters inside the EU. Connectivity to required third party tools is as good or even better than Azure DevOps. Runners can have similar capabilities compared to Azure DevOps. Organization structure might have to be different, but access to required data can be managed with organization, repository, team and user permissions.

4 GITHUB ACTIONS

GitHub Actions is a continuous integration and continuous delivery (CI/CD) platform that enables a way to automate software development processes. Automation is defined either in a workflow file or in a custom application file called action. Both are YAML files that can be used to automate many different software development tasks and processes including builds, deployments, testing, running scripts or triggering events in pre-defined conditions. YAML is a data serialization language and the acronym YAML comes from the words YAML Ain't Markup Language. GitHub Actions deployment features can be configured to use environments with specified protection rules and secrets only available to a given environment.

4.1 Workflow

Workflow YAML files must be defined in a specific directory called *.github/workflows*. It can be in repository that uses the workflow or a reusable workflow in other repository of the organization. One repository can have multiple workflow files, each assigned for different types of tasks. Workflows can be referenced in other workflows and chained to do different tasks for different events. (GitHub Inc, s.a.)

An event is a trigger that activates the workflow run, for example pull request, commit to specific branch or scheduled event. Workflow can contain multiple jobs that each can run in its own runner. Jobs can be executed either in sequential order or in parallel. Every job has a set of steps that are executed by the same runner. Steps are shell scripts or actions that are executed in order and can share data with each other. Each step is dependent on the previous steps so in case of failure the later steps won't run if some previous step fails. This will lead the whole job to be failed. Jobs can have conditional expression to run even if the previous job in a workflow fails. (GitHub Inc, s.a.)

The next figure shows an example workflow file with the following characteristics (Figure 2):

- Is named *example workflow.*
- Will be executed when someone pushes to main branch.
- Has four jobs and the first two run in Ubuntu and Windows operating systems.
- The second job is dependent on the first and will run after the first job.
- Steps uses an action called checkout with version 3.3.0 and runs a script in Bash or PowerShell.
- Has a job *call-workflow-file* that calls a workflow file called *second_workflow.yml* in the same repository.
- Has a job *call-reusable-workflow* that calls a workflow file called *reusable_workflow.yml* from the repository called *repo-name.*

```yaml
name: example workflow
on:
  push:
    branches:
      - main

jobs:
  ubuntu-job:
    runs-on: ubuntu-latest
    steps:
      - name: checkout
        uses: actions/checkout@v3.3.0

      - name: run a Bash script
        run: echo Hello from Bash!
        shell: bash

  windows-job:
    runs-on: windows-latest
    depends-on: First job
    steps:
      - name: checkout
        uses: actions/checkout@3.3.0

      - name: Run a Powershell script
        run: Write-Host Hello from Powershell!
        shell: pwsh

  call-workflow-file:
    name: execute another workflow file in the same repository
    uses: ./.github/workflows/second_workflow.yml

  call-reusable-workflow:
    name: call reusable workflow in another repository
    uses: org-name/repo-name/.github/workflows/reusable_workflow.yml@main
```

Figure 2 Example usage of triggers, jobs, steps, actions and workflow files calling in GitHub Actions workflow YAML file (Rautiainen, 2023)

Reusable workflows prevent copying and pasting the same automation tasks in the organization. They help maintaining workflow processes, promote best practices and new workflows can be created more quickly. Users who create workflows can concentrate on creating new tasks and reference previously done, well tested work of others and thus be more efficient. Parameters can be passed to workflows to customize the behavior. (GitHub Inc, s.a.)

Required workflows are pre-configured organization wide CI/CD policies. These workflows must run for all pull requests opened against the default branch. Unlike reusable workflows, they cannot be called in a workflow file. Required workflows are enforced to every old and new repository inside the organization. If the workflow result fails or the policies are not met, the pull request what triggered the workflow is not merged. Required workflows help to ensure quality and consistency in the development process. (GitHub Inc, s.a.)

4.2 Action

A custom application called action can be used when workflow uses frequently repeated tasks. Actions can be private self-made for the organization or public from GitHub Marketplace. There are three types of actions: Docker container action, JavaScript action and composite action. Common for all of these is the metadata file called action.yml that defines the inputs, outputs and the main entry point of the action. (GitHub Inc, s.a.)

Docker containers are customized environments for the actions code. Because they are already packaged with specific configuration, users do not have to install tools or manage dependencies in the runner. The only prerequisites are that the runner must have a Linux operating system with Docker installed and running. Docker containers are slower than other actions because of the time it takes to build the container environment. (GitHub Inc, s.a.)

JavaScript action separates code that action executes from the action.yml metadata file. Action should be written with pure JavaScript without dependencies to other binaries. JavaScript code is run directly on the runner and can use the binaries that the runner has. GitHub offers a toolkit repository to develop actions using Node.js. Node.js is a popular JavaScript runtime environment. It is an open-source and cross-platform and can be used to create network applications. (GitHub Inc, s.a.; OpenJS Foundation, s.a.).

Composite action is a bundle of workflow steps. They cannot have multiple jobs because they are the list of steps that use run keyword instead of job keyword. These differ from reusable workflows that can have multiple jobs. Composite actions are useful when there are the same workflow steps that run in many different workflows. Composite actions steps are all defined in action.yml file. (GitHub Inc, s.a.)

Versioning and release management of the actions is recommended if actions are used also by other users. This should be done with semantic versioning and tags. With tags the user of the action can easily decide in the workflow what major and minor version of the action is used. (GitHub Inc, s.a.)

4.3 Comparison to Azure DevOps

Azure DevOps and GitHub Actions are both very capable CI/CD platforms that automatically build, test, publish, release and deploy code. Both use a similar infrastructure as code (IaC) approach where pipelines and the workflows are defined in YAML files. Files include jobs that run parallel in separate virtual machines and can contain series of sequentially executed steps. These files can be reused in the organization to promote best practices and have centralized management of the workflows.

Platforms have some differences that must be taken into consideration. Azure DevOps can use the classic editor to configure the CI pipeline in GUI editor instead of YAML file. GitHub uses only the YAML file. Azure DevOps has a feature that can convert classic editor pipelines to corresponding YAML files.

YAML structure and keywords differ in some ways. The main difference is the use of stages in the Azure DevOps. GitHub Actions do not support stages and similar processes must be defined using separate workflow files. Actions are more precise of the structure of the YAML files, but in Azure DevOs some structure can be omitted.

Both platforms can use application components in their YAML files to perform a wide variety of re-used scenarios. These components are either self-made or publicly available from the platforms marketplace. In Azure DevOps these are called *tasks* and in Github Actions these are *actions*. GitHub has about 18 000 actions available in its marketplace and Azure DevOps has about 1500 tasks.

The following table compares some of the differences in context and keyword usage (Table 1).

Table 1. Concept and keyword changes in Azure DevOps and GitHub Actions

Concept	**Azure DevOps keyword**	**Concept**	**GitHub Actions keyword**
Pipeline		Workflow	
Trigger	trigger/pr/schedules	Event	on
Stage	stage	Not supported	
Job	job	Job	<job_id>
Dependency	demands	Dependency	needs
Agent	pool	Runner	runs-on
Task	task	Action	uses
Task parameters	inputs	Actions parameters	with
Conditions	condition	Conditions	if

4.4 Ponsse requirements

GitHub Actions is a CI/CD platform that seems to be fully capable of running the automated build, test and deployment workflows comparable to current Azure DevOps pipelines. The structure of the YAML files is different but the result should be the same with both services. Workflows can be triggered with required events like pull requests to a specific branch or after workflow runs. Deployments can be gated with environments and required checks. Self-hosted runners can have similar specifications as current ones. There were not found any missing features that would block the usage of GitHub Enterprise or GitHub Actions compared to Azure DevOps.

5 GITHUB ENTERPRISE SERVER DEMO INSTALLATION

GitHub Enterprise Server was deployed during this thesis (Figure 3). The installation was made into Azure Cloud services. The region for these installations was West Europe. Personal environments and subscriptions were used. Azure offers some services free for students so there were no costs for this setup. GitHub offered 45 days of free trial subscription of Enterprise accounts.

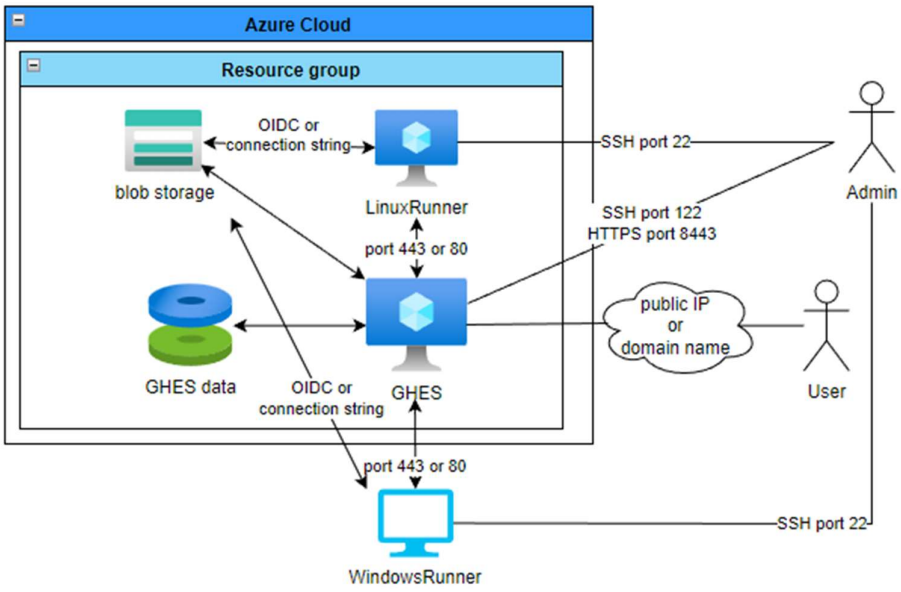

Figure 3 GitHub Enterprise Server demo installation architecture in Azure Cloud (Rautiainen, 2023).

The virtual machine size for the GHES instance was Standard E4s V3. This machine size was chosen because it met the minimum hardware requirements of the GHES and its usage costs were estimated to be as low as possible. Attached storage for GHES was Premium SSD with the size of 150 GB.

GHES must have the required network ports open in the firewall for administrative use and end user's services. Administrative ports are 122 for secure shell protocol (SSH) and 8443 for Hypertext Transfer Protocol Secure (HTTPS). The default SSH port number 22 is dedicated to Git version control and SSH application network traffic and it cannot be used for management of the GHES instance. End users access the services web application and Git over HTTPS in port 443 or HTTP in port 80.

5.1 Deployment

Deployment of the GitHub Enterprise Server instance was made with Azure CLI command interface following the GitHub's instructions for Azure deployment. The installation process was quite straight forward and the instructions were detailed enough. After the GHES instance installation was completed, Azure Portal was used to create a domain name for the instance. Initial setup process starts automatically when opening the GHES instance for the first time in the web browser. Initial setup process was done in GitHub Management Console and it includes multiple parts:

- Adding the GitHub Enterprise license file.
- Setting up a management console password.
- Choosing the installation type: new install, migrate or replica.
- Configuring basic settings such as domain name service (DNS), hostname and user authentication method, SSH keys for administrative SSH access to the instance.

Deployment is completed after the initial setup settings are saved and the instance is automatically rebooted. Now organizations and repositories can be created, users can be added and the instance can be used as a distributed version control system and for project management.

5.2 Adding GitHub Actions and runners

GitHub Actions feature was enabled in the GHES instance. Enabling is done in the Management Console. Actions require external storage for data generated in workflow runs. This data can be log files, caches and build artifacts. For this data, Azure blob storage was created to Azure Cloud and set to be the Actions storage in the Management Console. Blob storages are scalable and secure storages for unstructured data like text or binary data.

Two self-hosted runner machines were used for the GitHub Actions. One was a virtual machine with Linux operating system and was deployed to Azure Cloud. Another was a personal laptop with Windows 11 operating system. Installation requires access to the machine intended to be used as a runner environment and at least one organization must be created for the enterprise. Instructions for downloading, configuring and executing GitHub Actions Runner application are detailed commands for the command line interface. Instructions are found in the enterprise, organization or repository settings, depending on the scope runner is allowed to be used. Windows runner application was updated to the latest version, because the older version did not support all the new features in YAML syntax.

This GitHub Enterprise Server setup cost three to four dollars per day. The GHES instance and runner virtual machines were shut down when they were not used. Microsoft offers students 100 dollars free for Azure services, so it enabled about one month of free testing of GitHub Enterprise Server. Features and limitations compared to the GitHub.com version were easy to test and confirm using this setup. It really helped to learn management of the instance, the enterprise and the runners better than it would have been just by reading the documentation.

6 GITHUB ACTIONS IMPORTER

In March 2023 GitHub published Actions Importer to help plan and automate migration from different CI/CD platforms. This chapter goes through the main features of Actions Importer, basic usage with testing samples when migrating from Azure DevOps and conclusions and limitations that were found during the testing. It must be noted that Actions Importer is still new and many features were added even during this thesis. More features are also planned in the GitHub roadmap. (Gebregziabher, 2023; GitHub Inc, s.a.)

Testing was done with simplified pipelines compared to actual pipelines of Ponsse's projects but tried to be created so that they highlight possible limitations of the tool. Testing environments were personal Windows 11 laptop and GitHub Codespaces development environment. Azure DevOps and GitHub accounts were the author's personal with free student subscription.

Actions Importer can be used to migrate pipelines from Azure DevOps, CircleCI, GitLab, Jenkins and Travis CI platforms. It is free to use for anyone under the MIT license and the source code can be found from *github.com/github/gh-actions-importer* repository. The distribution method for Action Importer is a Docker container and extension of the GitHub CLI command interface is used for interactions. Requirements for usage are an environment which has Docker installed and running and GitHub CLI installed. The environment can be a local computer or a virtual machine. Users must have a GitHub account and access to GitHub Enterprise Server and Azure DevOps Personal Access Tokens. The goal for the tool is to convert 80 percent of the workflows automatically. (GitHub Inc, s.a.)

By default, the tool collects anonymous telemetry data of the usage. Using the **`--no-telemetry`** flag in the GitHub CLI commands disables data collection.

6.1 Installation and configuration

The installation is done with the GitHub CLI command:

```
$ gh extension install github/gh-actions-importer
```

The latest version can be updated with the command:

```
$ gh actions-importer update
```

To communicate with GitHub Enterprise Server and the Azure DevOps the tool must be configured with credentials to the .env.local file or to environment variables with interactive configuration setup. Tool uses personal access token (PAT) for credentials, that must be created in both platforms. PATs must have required scopes when created. GitHub scope must be set as workflow. Azure DevOps scopes are reading rights to Agent pool, Build, Code, Release, Service Connections, Task Groups and Variable Groups. (GitHub Inc, s.a.)

The configuration is done with the command:

```
$ gh actions-importer configure
```

The configuration command opens an interactive configuration setup to the command interface and guides through the setup of credentials to different platforms (Figure 4).

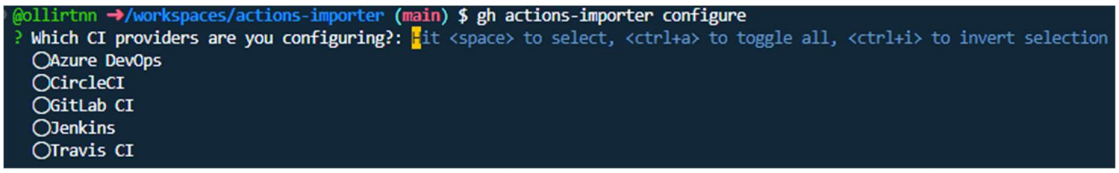

Figure 4 Actions Importer Configure command and interactive configuration setup (Rautiainen, 2023)

After the configuration setup is completed, .env.local file is created to the root folder (Figure 5). It can be modified directly if necessary. Because .env.local file contains personal access tokens in plain text, it should be confirmed that the file is mentioned in .gitignore file to ensure it is not pushed to version control. Access tokens in the figure are no longer active.

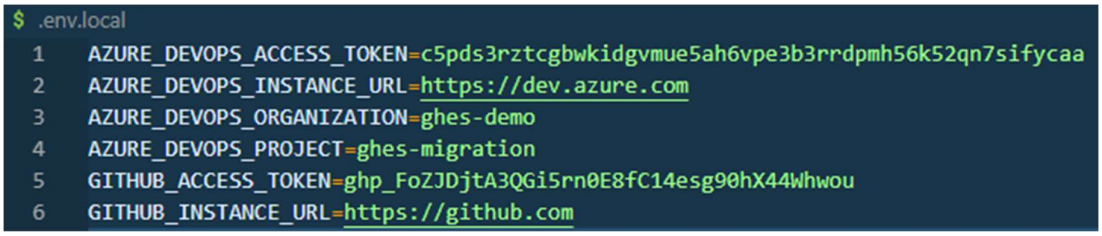

Figure 5 Content of the .env.local file after the completed Actions Importer configuration setup (Rautiainen, 2023).

6.2 Audit

Audit feature is executed with GitHub CLI command:

```
gh actions-importer audit <platform> --output-dir <directory>
```

Audit command creates a wide view of the pipelines used in every project in the Azure DevOps organization. It fetches all the pipelines in the projects, converts them to equal GitHub Actions workflow files and generates a summary report of how complete the transform would be. It also creates a workflow usage report file. (Figure 6). For example, to perform an audit command for Azure DevOps project that is defined in .env.local file and output the files to directory ./projects/ghes-migration, the command would be:

```
gh actions-importer audit azure-devops –output-dir projects/ghes-migration
```

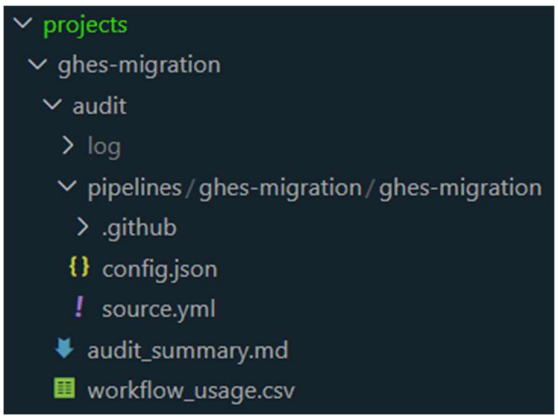

Figure 6 Generated files after successful Actions Importer audit command (Rautiainen, 2023).

Audit report is called audit_summary.md (Figure 7). It contains information about pipelines, job types, build steps, triggers, environments and manual tasks and links to generated files. It tries to sum up the success of the migration and help to visualize what can automatically be migrated. For each pipeline file, the audit also creates additional files such as the original pipeline in Azure DevOps, network responses in conversion and stack traces to help troubleshoot possible issues in conversion.

```
# Audit summary

Summary for [Azure DevOps instance](https://dev.azure.com/ghes-demo/ghes-migration/_build)

- GitHub Actions Importer version: **1.1.16912 (35f586628531ad7d7f0e772abeeda681da325bd4)**
- Performed at: **3/21/23 at 14:04**

## Pipelines

Total: **1**

- Successful: **1 (100%)**
- Partially successful: **0 (0%)**
- Unsupported: **0 (0%)**
- Failed: **0 (0%)**

### Job types

Supported: **1 (100%)**

- YAML: **1**

### Build steps

Total: **12**

Known: **12 (100%)**

- NuGetToolInstaller@1: **2**
- VSBuild@1: **2**
- checkout: **1**
- PowerShell@2: **1**
- powershell: **1**
- myAction@6(custom): **1**
- InlinePowershell@1: **1**
- VSTest@2: **1**
- NuGetCommand@2: **1**
- VisualStudioTestPlatformInstaller@1(custom): **1**

Actions: **17**

- run: **6**
```

Figure 7 Partial content of the audit_summary.md file (Rautiainen, 2023).

Workflow_usage.csv file shows all the created actions, secrets and runners and lists them to the name and path to the pipeline they are used on (Figure 8). It helps to visualize what different workflows are used, for example for security reviews.

```
Pipeline,Action,File path
ghes-migration/ghes-migration,azure/login@v1.4.6,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml
ghes-migration/ghes-migration,actions/checkout@v3.3.0,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml
ghes-migration/ghes-migration,nuget/setup-nuget@v1.1.1,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml
ghes-migration/ghes-migration,microsoft/setup-msbuild@v1.3.1,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml
ghes-migration/ghes-migration,microsoft/vstest-action@v1.0.0,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml
ghes-migration/ghes-migration,./.github/actions/templates_template1,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml
ghes-migration/ghes-migration,./.github/actions/templates_more_template3,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml
ghes-migration/ghes-migration,./.github/workflows/templates_template2.yml,projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml

Pipeline,Secret,File path
ghes-migration/ghes-migration,${{ secrets.AZURE_CREDENTIALS }},projects/ghes-migration/audit/pipelines/ghes-migration/ghes-migration/.github/workflows/ghes-migration.yml

Pipeline,Runner,File path
gh-importer-labs/pipelines/pipeline2,mechamachine,tmp/audit/pipelines/gh-importer-labs/pipelines/pipeline2/.github/workflows/pipeline2.yml
```

Figure 8 Content of the workflow_usage.csv file (Rautiainen, 2023).

6.3 Forecast

Forecast feature is executed with GitHub CLI command:

```
gh actions-importer forecast <platform> --output-dir <directory>
```

Forecast command creates forecast_report.md file. It lists the pipeline usage in Azure DevOps and helps to predict what the usage would be in GitHub. It shows the total number of completed jobs, number of unique pipelines and execution, queue and concurrent job times. (Figure 9).

```
## Azure Pipelines

- Job count: **18**
- Pipeline count: **1**

- Execution time

  - Total: **38 minutes**
  - Median: **2 minutes**
  - P90: **3 minutes**
  - Min: **0 minutes**
  - Max: **3 minutes**
```

Figure 9 Partial content of the forecast_report.md file (Rautiainen, 2023).

6.4 Dry-run

Dry-run feature is executed with GitHub CLI command:

```
gh actions-importer dry-run <platform> pipeline --pipeline-id <pipeline-id> --output-dir <directory>
```

Dry-run tries to convert specific Azure DevOps pipeline to comparable GitHub Actions workflow. Files are created to the directory specified in --output-dir flag. Conversion is done to one pipeline at a time because the pipeline ID must be specified. The pipeline ID can easily be found by navigating to Azure DevOps pipeline and the browser's address bar specifies the pipelines *definitionId* in the address. Dry runs can be performed to build and release pipelines.

For example, to perform a dry-run command for Azure DevOps project defined in .env.local file, specify the pipeline with an Id value of 1 and output the files to the directory ./projects/ghes-migration/dry-run, the command would be:

```
gh actions-importer dry-run azure-devops pipeline --pipeline-id 1 --
output-dir projects/ghes-migration/dry-run
```

Successful dry-run automatically converted Azure DevOps pipeline and template files to the equivalent workflow and actions files depending on how these files were used in the pipeline (Figure 10). Step templates in Azure DevOps are converted to composite actions. Job and stage templates and templates that are extending from another template are converted into reusable workflows.

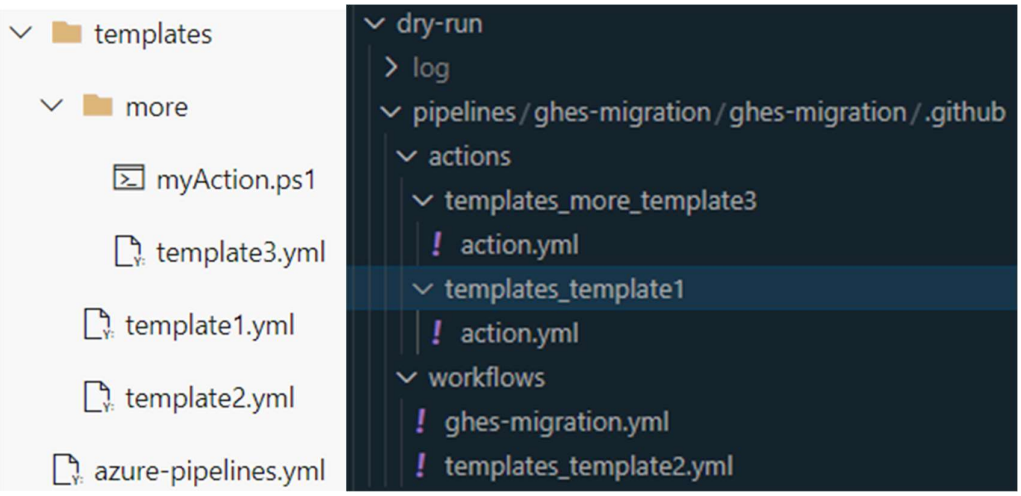

Figure 10 Left: Azure DevOps pipeline and template files. Right: The equivalent workflow and action files after dry-run command (Rautiainen, 2023).

Template2.yml file is under a job key that runs template in the pipeline (Figure 11). That is why it is converted to a standalone reusable workflow file. The same would be done to templates used under stage or deployment keys.

```yaml
- stage: run_template2
  displayName: run template2
  dependsOn:
  - build
  jobs:
  - template: templates/template2.yml
    parameters:
      TargetBranch: 'main'
```

Figure 11 Use of template2.yml file in Azure DevOps pipeline (Rautiainen, 2023).

Files template1 and template3 are step templates so these are converted to composite actions (Figure 12). The importer tool automatically creates folders to *./.github/actions* folder and the relevant action.yml files.

```yaml
jobs:
- job: Build
  displayName: Build
  steps:
  - template: templates/template1.yml

  - template: /templates/more/template3.yml
```

Figure 12 Use of template1 and template3 in the Azure DevOps pipeline (Rautiainen, 2023).

6.5 Migrate

Migrate feature is executed with GitHub CLI command:

```
gh actions-importer migrate <platform> pipeline --pipeline-id <pipeline id> --target-url <github-repository-url> --output-dir <log-directory>
```

The migrate command makes the same conversion to pipelines as the dry-run command, but also opens a pull request to the target repository in GitHub. The target repository must already be created to GitHub before the migration. The output directory determines where the log files will be put. For example, to migrate pipelines to GitHub repository called *actions-importer* in the GitHub organization called *ollirtnn-playground-org*, the command would be:

```
gh actions-importer migrate azure-devops pipeline --pipeline-id 1 --target-url https://github.com/ollirtnn-playground-org/actions-importer --output-dir projects/ghes-migration/migrate
```

Log files are stored into the folder *./projects/ghes-migration/migrate*. Pull request is opened to the base branch in GitHub repository (Figure 13).

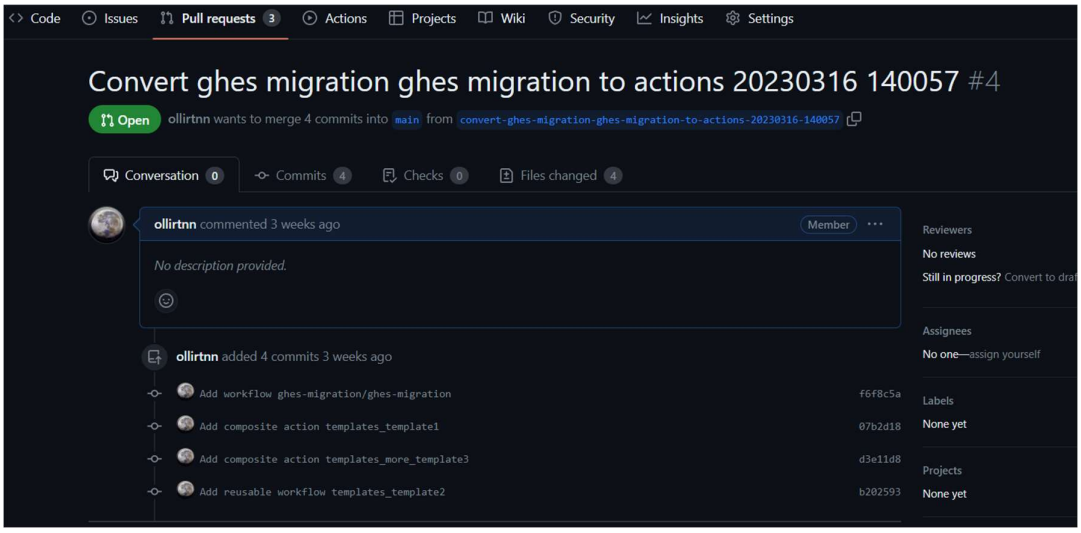

Figure 13 Pull request in GitHub made by the migration command (Rautiainen, 2023).

The files and the folder structure are the same as in the dry-run command (Figure 14). The pull request can now be easily inspected, commented and reviewed. After the pull request changes are merged, the workflow and action files are automatically added to the repository's Actions menu.

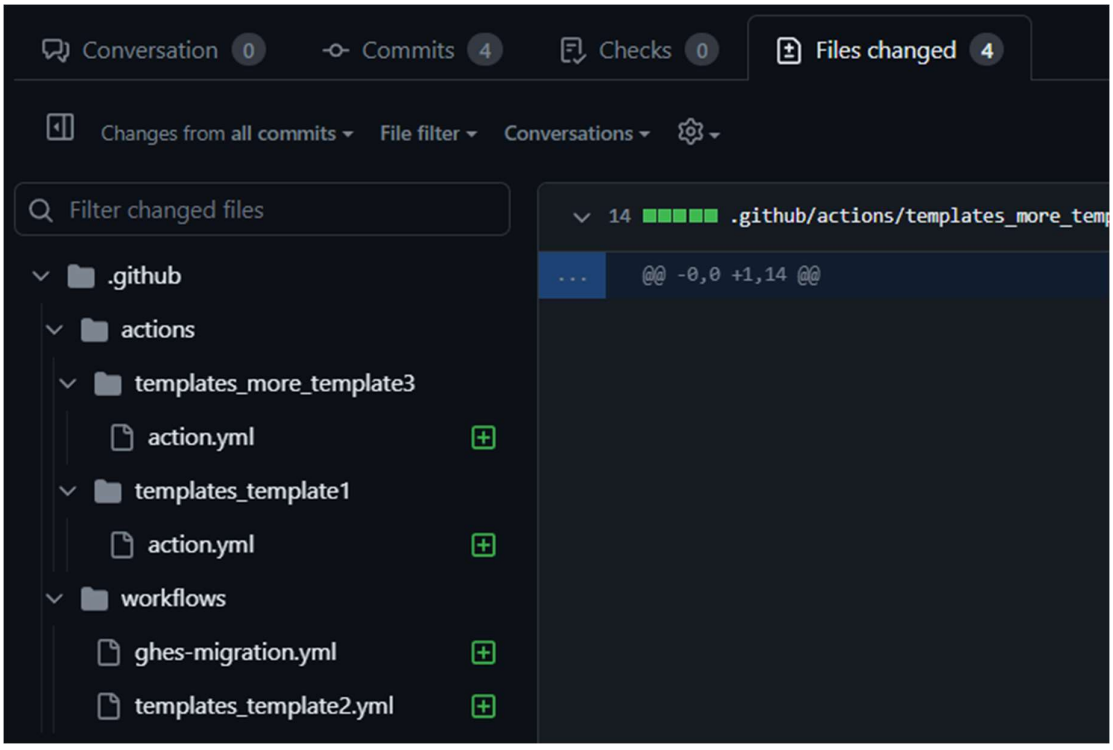

Figure 14 Files changed tab in the pull request made by the migration command (Rautiainen, 2023).

6.6 Custom transformers

Some features in Azure DevOps pipelines cannot be converted automatically. Also, in some situations the user might want to alter the behavior of the conversion. Actions Importer can use custom transformer files to extend its basic functions. They are files that contain mapping logic that the tool can use to transform steps, map environment variables, change script handling and so on. Files are domain specific language (DSL) built on top of Ruby, have .rb extension and must locate in the same directory or subdirectory where the tool commands are run. Custom transformers are used with --custom-transformers CLI option followed by the filename of the transformer file or glob pattern syntax for multiple files. For example, to execute the dry-run command with the transformer file called transformers.rb, the command would be:

```
gh actions-importer dry-run azure-devops pipeline --pipeline-id 1 --
output-dir projects/ghes-migration/dry-run --custom-transformers
transformers.rb
```

Custom transformers can for example map environment variables to specific values or GitHub secrets or remove them completely. Runners can be re-labeled to one or multiple labels. The default runner value in Importer is ubuntu-latest if the runner is not recognized correctly. This default behavior can also be changed to point to a different runner label. (Figure 15).

```
1  env "BUILDCONFIGURATION", "Debug"
2  env "isMain", "${{ github.ref == 'refs/heads/main' }}"
3  runner :default , "my-runnergroup"
```

Figure 15 In custom transformer file the environment variables *BUILDCONFIGURATION* and *isMain* are mapped to new values. The default runner label for runners that are not automatically converted is changed to *my-runnergroup* (Rautiainen, 2023).

Items can be converted if Importer does not recognize them automatically. In this example VisualStudioTestPlatformInstaller@1 Azure DevOps task is not recognized by the Importer so transformer is used to change it to the corresponding .NET CLI command (Figure 16; Figure 17).

```
- task: VisualStudioTestPlatformInstaller@1
  displayName: 'Visual Studio Test Platform Installer'
  inputs:
    testPlatformVersion: '$(testPlatformVersion)'
```

Figure 16 Example of Azure DevOps task that Actions Importer cannot convert automatically (Rautiainen, 2023).

```
transform "VisualStudioTestPlatformInstaller@1" do |item|
  {
    run:   "dotnet add package Microsoft.TestPlatform --version #{item['testPlatformVersion']}",
    shell: "pwsh",
  }
end
```

Figure 17 In custom transformer file the unrecognized Azure DevOps task is converted into corresponding .NET CLI command (Rautiainen, 2023).

6.7 Required manual work

Testing concluded that Actions Importer automatically converts a large part of the Azure DevOps pipelines and is an essential tool when planning, testing and implementing the migration. There are many manual tasks to be taken into consideration but nevertheless the work amount is reduced significantly. Most time-consuming work would be the manual conversion of the unrecognized Azure DevOps build-in and custom tasks. Actions Importer cannot convert all the built-in tasks to GitHub Actions or conversion does not support all the features. All the supported tasks and their possible unsupported inputs can be found from the *github.com/github/gh-actions-importer/docs/azure_devops* repository from GitHub. (GitHub Inc, s.a.)

Custom tasks are not converted at all. These must be handled manually with transformer files or done separately to GitHub. Other manual tasks are:

- Add variables and secrets to GitHub.
- Add runners to GitHub and re-label in workflow if needed.
- Some custom condition expressions are not converted.
- Inline PowerShell steps are not always converted correctly.
- Needed actions must be added to GHES if GitHub Connect is not used.

The way to handle these tasks should be planned individually based on their usage rate in pipelines, complexity and properties. The good thing is that both the Actions Importer and GitHub Enterprise Server are evolving and updated regularly. Many features were improved and added during this thesis and the roadmap has a lot of improvements planned. The documentation is also regularly updated.

7 GITHUB COPILOT

GitHub Enterprise subscription enables the possibility of the use of the GitHub Copilot for Enterprise members. This chapter reviews GitHub Copilot as a business user privacy point of view.

GitHub Copilot is an artificial intelligence (AI) pair programmer that can auto-complete comments and code. It uses Codex learning model created by the company OpenAI. Codex data is collected from publicly available sources like public repositories on GitHub. Copilot has two generally available versions of the product, Copilot for Individuals and Copilot for Business. (GitHub Inc, s.a.)

Basic functions of both versions are:

- Integration directly to multiple code editors.
- Turn natural language writing to code.
- Creation of multi-line suggestions.
- Creation of tests for the code.
- Filter suggestions that match the public code.

Copilot For Business extends basic functions with:

- License and policy management.
- Stricter data collection.
- Support for virtual private network (VPN) proxy.

Enabling Copilot for Business requires purchase of the Copilot subscription and configurations of enterprise or organization level policies and setting. Enterprise owners must determine which organizations are allowed to use Copilot and allow or block the use of public code in suggestions. Individual people or teams can then be assigned to have Copilot seats for the Copilot subscription. To use Copilot, the extension must be installed to the integrated development environment (IDE). GitHub does not own the code it suggests and the responsibility to verify, test and use the suggested code is with the user. (GitHub Inc, 2023)

7.1 Privacy of Copilot for Business

GitHub does not collect code data from Business customers. Copilot uses the file content and user engagement data to work. Suggestions are made based on made comments or a small part of the source code called code snippets. These snippets are transmitted from IDE to GitHub for processing. The connection is encrypted and done in real-time. The code snippets are discarded after the suggestion is returned to IDE. This differs from the Copilot for Individuals where the code snippets are retained and used as part of the Codex learning data and can be accessed by the GitHub, Microsoft or OpenAI personnel. (GitHub Inc, s.a.)

GitHub collects user engagement data from Business customers. User engagement data includes information on how the IDE is interacted by the user and used to provide service and for Copilots product development and research. Data is collected by GitHub and is shared with Microsoft and OpenAI. Data collection is required for the use of Copilot and neither Individual nor Business customers can opt-out from this. (GitHub Inc, s.a.)

Collected user engagement data includes:

- Accepted or dismissed Copilot suggestions.
- Error and latency data.
- Feature engagement data.
- Personal data as pseudonymous identifiers.

Pseudonymisation means that personal data is processed in such a way that that it can no longer be linked to a specific person. Individuals from pseudonymized personal data can be identified but it requires additional information. (Office of the Data Protection Ombudsman, s.a.)

7.2 Copilot X

GitHub is developing a new version of Copilot called Copilot X. It is currently in technical preview and not publicly available. (Dohmke, 2023). Privacy aspects should be reviewed after it has been published.

Copilot X expands from just auto-completing code to a chat like experience. Copilot Chat uses OpenAI's new GPT-4 model and can create in-depth analysis about the code, generate unit tests and propose bug fixes for the code. It also has an accessibility function to generate code from voice. GitHub states that Copilot for Individuals and Businesses will stay as their own versions after Copilot X is published and with the same features as they have now. (GitHub Inc, s.a.)

8 CONCLUSION

The aim of the thesis was to research the state of the GitHub Enterprise platform mainly from the CI/CD features point-of-view. Find out GitHub Actions CI/CD platform capabilities compared to Azure DevOps and the amount of the work what migration process of the current pipelines requires. This thesis also reviews the current privacy state of the GitHub Copilot. The process of the thesis started by researching different internet sources of similar comparisons for Azure DevOps. Quickly, it was clear that most of these sources were out of date because GitHub Enterprise has had rapid release state for the past few years. So, to have as accurate data as possible, most of the data was either tested and confirmed in demo environments or relied on GitHub documentation. GitHub Enterprise Server with GitHub Actions features was deployed to Azure using the free student benefits of Azure and free trial license of GitHub Enterprise.

GitHub Enterprise has two versions, GitHub hosted Enterprise Cloud and self-hosted Enterprise Server. Server version is compliant with Ponsse's requirements for EU based deployment locations and self-hosted instance and runner management either in on-premises, in virtualization hypervisors or with required cloud providers. GitHub Actions is a continuous integration and continuous delivery (CI/CD) platform. It can automate build, test and deployment pipelines with highly customizable workflows. Workflows can also automate many other tasks and processes, react to different events automatically or be triggered manually. Compared to Azure DevOps the difference in CI/CD capabilities between these platforms is minimal. Azure DevOps might have more capabilities in testing and project management services, but the state of these features was not tested during this thesis work. GitHub has improved its Enterprise and Actions capabilities and features rapidly in the past few years and there are several improvements in the GitHub roadmap planned related to these products.

GitHub Actions and Azure DevOps have differences in syntax of the YAML file that defines the pipeline. Pipeline files cannot be copied directly between platforms. This thesis covers the usage and limitations of the latest migration tool GitHub Actions Importer. It is a tool designed to help plan and automate migration from different CI/CD platforms. Testing of the tool was done in demo environments with sample pipelines. Actual pipelines are far more complex so the migration success rate will be revealed only after it is run against real pipelines. Still, the benefits of the migration tool were clearly visible. It makes planning and management of the migration process easier and more predictable. It can be estimated that the most time-consuming manual work of the migration will be the conversion of the Azure DevOps tasks that Actions Importer cannot recognize. All the self-built custom tasks must be converted manually. Other manual tasks are adding variables, secrets and runners to GitHub and to the workflow files. Actions Importer was not always able to convert condition expressions or inline shell scripts correctly, so there is a high chance that there will be manual work even if the conversion process runs successfully. GitHub Enterprise Server comes with just the basic actions pre-installed, so most of the required actions must be installed to the server instance manually or GitHub Connect service must be enabled for automatic connection to GitHub.com.

The thesis reviewed the current privacy state of GitHub Copilot. The review was done to available versions Copilot for Individuals and Copilot for Business. The latest version Copilot X is currently only in technical preview, so the privacy state of this service was not reviewed. GitHub states that it does

not collect code data from Copilot for Business users. The code snippets data that the service uses for creating suggestions is handled encrypted in real time and is not retained by GitHub. Code of the Copilot for Business users is not used in other Copilot users' suggestions. GitHub defines Copilot as a tool and does not own the code that Copilot generates. Data that GitHub does collect from all users is user engagement data. This data is used for product improvements and research purposes and shared with Microsoft and OpenAI. Data includes information on how the user interacts with the Copilot suggestions, error and latency data and how the features are engaged by the user. Personal data is collected as pseudonymous identifiers and cannot be linked to a specific user.

9 DISCUSSION

Software development projects are more dependent on automation than ever before. Code bases are large and the whole is hard for humans to understand and manage. Automation, machine learning and artificial intelligence products and features help tremendously in managing the development cycle and keeping developer's workflow as productive as possible.

GitHub Enterprise has many features that are designed to help developers and administrators in their work. Both Microsoft owned platforms Azure DevOps and GitHub Enterprise are currently capable of similar end-results, even though with a little different approach in workflows and in terminology. GitHub Copilot extends automation to developer workflow with artificial intelligence powered code suggestions. Microsoft has revealed to publish Copilot features also to its Microsoft 365 and Dynamics products.

Tools and features, like Actions Importer, for migrating between Azure DevOps and GitHub Enterprise are developed and improved continuously. Even though the migration still requires a lot of manual transformation and verification, tools are valuable assets when planning the migration processes, finding out required tasks and evaluating the schedule of the migration process.

Thesis work was done between January and April 2023. Thesis process went almost as planned. The amount of new or improved features published just in the middle of this work was a little surprising and made the time management and planning difficult. Actions Importer tool was published in March and was such an important feature that it had to be reviewed thoroughly to get relevant up-to-date results for the research questions of this thesis.

Working on this thesis helped me to understand better the CI/CD pipelines features and the complexity of the automation processes. It improved my understanding of the YAML pipelines syntax and the use of different techniques in the pipelines. I will continue to work in Ponsse DevOps CICD team after this thesis is completed and even though the GitHub is not used in my work, many of the concepts can be applied universally also in Azure DevOps and broader in DevOps work. I find it beneficial to have a broad view of different possibilities and ways in which different tasks can be done. It helps to set standards, review the quality of your work and make improvements to work methods.

This thesis covered mostly just the CI/CD features of the GitHub Enterprise. The migration of the pipelines is just one part of the whole implementation of GitHub Enterprise platform. Server installation to Ponsse environments, subscription and access management, security of the platform and security products it offers and cost comparisons are some of the topics for future work. The pipeline migration process should be tested further with Actions Importer to real pipelines for a more precise estimate of the required workload. Privacy and security concerns of GitHub Copilot and other AI-powered products are relevant. GitHub and Microsoft do underline the safety of the data when marketing these products. Microsoft is bringing Copilot features also to its own business user focused products like Power Platform and Teams, so soon there might not even be an option not to have any AI features integrated in Microsoft products.

REFERENCES

Boyle, Jim and Moths, Jessi. 2023. Level up monitoring and reporting for your enterprise. [Online] April 3, 2023. [Cited: April 17, 2023.] https://github.blog/2023-04-03-level-up-monitoring-and-reporting-for-your-enterprise/.

Colyer, Matt. 2016. GitHub Enterprise 2.5 is now available. [Online] February 9, 2016. [Cited: April 17, 2023.] http://www.bluecomtech.com/Web%20Sites/saltfactory/github.com/blog/category/enterprise.html.

DevOps Research and Assessment. 2022. Announcing the 2022 Accelerate State of DevOps Report: A deep dive into security. [Online] September 29, 2022. [Cited: April 17, 2023.] https://cloud.google.com/blog/products/devops-sre/dora-2022-accelerate-state-of-devops-report-now-out.

Dohmke, Thomas. 2023. 100 million developers and counting. *github.blog.* [Online] January 25, 2023. [Cited: April 17, 2023.] https://github.blog/2023-01-25-100-million-developers-and-counting/.

—. 2023. GitHub Copilot X: The AI-powered developer experience. [Online] 3. 22, 2023. [Cited: 4. 18, 2023.] https://github.blog/2023-03-22-github-copilot-x-the-ai-powered-developer-experience/.

Finnish Transport and Communications Agency, National Cyber Security Centre, Traficom. 2021. Riippuvuusristiriidat altistavat hyökkäyksille. [Online] 10. 29, 2021. [Cited: 4. 17, 2023.] https://www.kyberturvallisuuskeskus.fi/fi/ajankohtaista/riippuvuusristiriidat-altistavat-hyokkayksille.

Gebregziabher, Dawit. 2023. GitHub Actions Importer is now generally available. *The GitHub Blog.* [Online] 3. 1, 2023. [Cited: 4. 18, 2023.] https://github.blog/2023-03-01-github-actions-importer-is-now-generally-available/.

GitHub Inc. s.a.. About clustering. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/enterprise-management/configuring-clustering/about-clustering.

—. s.a.. About custom actions. [Online] s.a. [Cited: 4. 18, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/creating-actions/about-custom-actions.

—. s.a.. About Dependabot alerts. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/code-security/dependabot/dependabot-alerts/about-dependabot-alerts.

—. s.a.. About enterprise accounts. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/overview/about-enterprise-accounts.

—. s.a.. About GitHub AE. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/github-ae@latest/admin/overview/about-github-ae.

—. s.a.. About GitHub Connect. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/configuration/configuring-github-connect/about-github-connect.

—. s.a.. About GitHub Enterprise Cloud. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-cloud@latest/admin/overview/about-github-enterprise-cloud.

—. s.a.. About GitHub Enterprise Server. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/overview/about-github-enterprise-server.

—. s.a.. About GitHub for enterprises. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/overview/about-github-for-enterprises.

—. s.a.. About GitHub-hosted runners. [Online] s.a. [Cited: Aprl 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/using-github-hosted-runners/about-github-hosted-runners.

—. s.a.. About high availability configuration. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/enterprise-management/configuring-high-availability/about-high-availability-configuration.

—. s.a.. About teams. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/organizations/organizing-members-into-teams/about-teams.

—. s.a.. About ugrades to new releases. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/overview/about-upgrades-to-new-releases.

—. s.a.. About workflows. [Online] s.a. [Cited: 4. 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/using-workflows/about-workflows.

—. s.a.. Accessing the monitor dahboard. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/enterprise-management/monitoring-your-appliance/accessing-the-monitor-dashboard.

—. s.a.. Automating migration with GitHub Actions Importer. [Online] s.a. [Cited: 4. 18, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/migrating-to-github-actions/automated-migrations/automating-migration-with-github-actions-importer.

—. s.a.. Best practices for structuring organizations in your enterprise. [Online] s.a. [Cited: 4. 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/user-management/managing-organizations-in-your-enterprise/best-practices-for-structuring-organizations-in-your-enterprise.

—. s.a.. Creating a composite action. [Online] s.a. [Cited: 4. 18, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/creating-actions/creating-a-composite-action.

—. s.a.. Creating a Docker container action. [Online] s.a. [Cited: 4. 18, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/creating-actions/creating-a-docker-container-action.

—. s.a.. GitHub Copilot - Your AI pair programmer. [Online] s.a. [Cited: 4. 18, 2023.] https://github.com/features/copilot/.

—. s.a.. GitHub Copilot for Business Privacy Statement. [Online] s.a. [Cited: 4. 18, 2023.] https://docs.github.com/en/site-policy/privacy-policies/github-copilot-for-business-privacy-statement.

—. 2023. GitHub Copilot Product Specific Terms. [Online] 3. 2023. [Cited: 4. 18, 2023.] https://github.com/customer-terms/github-copilot-product-specific-terms.

—. 2020. GitHub public roadmap. [Online] July 24, 2020. [Cited: April 17, 2023.] https://github.com/github/roadmap/issues/72.

—. s.a.. GitHub public roadmap. [Online] s.a. [Cited: 4. 18, 2023.] https://github.com/orgs/github/projects/4247.

—. s.a.. github/gh-actions-importer repository. *GitHub.* [Online] s.a. [Cited: 4. 18, 2023.] https://github.com/github/gh-actions-importer/tree/main/docs/azure_devops.

—. s.a.. Introducing GitHub Copilot X. [Online] s.a. [Cited: 4. 18, 2023.] https://github.com/features/preview/copilot-x.

—. s.a.. Migrating from Azure DevOps with GitHub Actions Importer. [Online] s.a. [Cited: 4. 18, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/migrating-to-github-actions/automated-migrations/migrating-from-azure-devops-with-github-actions-importer.

—. s.a.. Releasing and maintaining actions. [Online] s.a. [Cited: 4. 18, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/creating-actions/releasing-and-maintaining-actions.

—. s.a.. Required workflows. [Online] s.a. [Cited: 4. 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/using-workflows/required-workflows.

—. s.a.. Reusing workflows. [Online] s.a. [Cited: 4. 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/actions/using-workflows/reusing-workflows.

—. s.a.. Roles in an enterprise. [Online] s.a. [Cited: April 17, 2023.] https://docs.github.com/en/enterprise-server@3.8/admin/user-management/managing-users-in-your-enterprise/roles-in-an-enterprise.

GitHub, Inc. s.a.. Enterprise: A smarter way to work together. [Online] s.a. [Cited: April 17, 2023.]

Kalliamvakou, Eirini. 2022. Research: quantifying GitHub Copilot's impact on developer productivity and happiness. *GitHub.blog.* [Online] September 7, 2022. [Cited: April 17, 2023.]

https://github.blog/2022-09-07-research-quantifying-github-copilots-impact-on-developer-productivity-and-happiness/.

Kinsbruner, Eran. 2020. *Accelerating Software Quality, Machine Learning and Artificial Intelligence in the Age of DevOps.* s.l. : Perforce Software, Inc, 2020. ISBN 9798671126044.

Lardinois, Frederic. 2022. Four years after being acquired by Microsoft, GitHub keeps doing its thing. *TechCrunch Web site.* [Online] October 26, 2022. [Cited: April 17, 2023.] https://techcrunch.com/2022/10/26/four-years-after-being-acquired-by-microsoft-github-keeps-doing-its-thing/.

Office of the Data Protection Ombudsman. s.a.. Pseudonymised and anonymised data. [Online] s.a. [Cited: 4. 19, 2023.] https://tietosuoja.fi/en/pseudonymised-and-anonymised-data.

OpenAI Corporation. s.a.. OpenAI. [Online] s.a. [Cited: April 17, 2023.] https://openai.com/.

OpenJS Foundation. s.a.. Introduction to Node.js. [Online] s.a. [Cited: 4. 18, 2023.] https://nodejs.dev/en/learn/.

Ponsse Plc. s.a.. Business concept. [Online] s.a. [Cited: April 17, 2023.] https://www.ponsse.com/company/ponsse/business-concept#/.

—. 2023. Ponsse's Financial Statements for 1 January - 31 December 2022. [Online] February 21, 2023. [Cited: April 17, 2023.] https://mb.cision.com/Public/18192/3719699/895e8856a481f849.pdf.

Rautiainen, Olli. 2023. *.env.local file after the completed Actions Importer configuration setup.*

—. 2023. *Actions Importer Configure command and interactive configuration setup.*

—. 2023. *Content of the workflow_usage.csv file.*

—. 2023. *Example of Azure DevOps task that Actions Importer cannot convert automatically.*

—. 2023. *Example usage of triggers, jobs, steps, actions and workflow files calling in GitHub Actions workflow YAML file.*

—. 2023. *Files changed tab in the pull request made by the migration command.*

—. 2023. *Generated files after successful Actions Importer audit command .*

—. 2023. *GitHub Enterprise Server demo installation architecture in Azure Cloud.*

—. 2023. *In custom transformer file the environment variables BUILDCONFIGURATION and isMain are mapped to new values. The default runner label for runners that are not automatically converted is changed to my-runnergroup .*

—. 2023. *Left: Azure DevOps pipeline and template files. Right: The equivalent workflow and action files after dry-run command .*

—. 2023. *Partial content of the audit_summary.md file .*

—. 2023. *Partial content of the forecast_report.md file.*

—. 2023. *Pull request in GitHub made by the migration command.*

—. 2023. *Sef-hosted runer configuration experience in Ubuntu virtual machine.*

—. 2023. *Unrecognized Azure DevOps task conversion to corresponding .NET CLI command in custom transformer file.*

—. 2023. *Use of template1 and template3 in the Azure DevOps pipeline.*

—. 2023. *Use of template2.yml file in Azure DevOps pipeline.*

Schneider, Lars. 2020. Hardening your GitHub Enterprise Server. [Online] July 20, 2020. [Cited: April 17, 2023.] https://github.blog/2020-07-20-hardening-your-github-enterprise-server/.

Stallbaumer, Colette. 2023. Introducing Microsoft 365 Copilot—A whole new way to work. [Online] March 16, 2023. [Cited: April 17, 2023.] https://www.microsoft.com/en-us/microsoft-365/blog/2023/03/16/introducing-microsoft-365-copilot-a-whole-new-way-to-work/.

Thomas, Daniel. 2022. What Log4Shell taught us about application security, and how to respond now. *SC magazine.* [Online] 7. 5, 2022. [Cited: 4. 17, 2023.] https://www.scmagazine.com/resource/application-security/what-log4shell-taught-us-about-appsec-and-how-to-respond.