Tuomas Hentonen

AUTOMATED SETUP OF TEST ENVIRONMENTS

Master's Degree Programme in Information Technology

2019

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

AUTOMATED SETUP OF TEST ENVIRONMENTS

Hentonen, Tuomas
Satakunta University of Applied Sciences
Degree Programme in Information Technology.
April 2019
Supervisor: Kyngäs, Jari
Number of pages: 33
Appendices: -

Keywords: test environment, automation, virtualization, testing

---

Companies are creating software for many different purposes. This software needs to be tested. For this purpose, a test environment is used. The setup of the test environment – on-site or cloud-based takes a certain amount of time and resources from the company. The traditional method to do the setup is to create a template which will be cloned to all of the test servers and afterwards all configuration is done manually.

Commit has many projects each year where a test environment is used. The setup of test environments takes a lot of resources in the company and because of this there is a need to find a way to automatize the setup process. The solution needs to be easy to use for each team member at Commit.

The research type used in this work was action research which is a suitable method for improving an already existing process.

The thesis will be most beneficial to Commit, which will get a faster process and a proper documentation for starting up their virtualized test environments. The thesis includes general information about test environments and test processes at Commit. A short comparison between on-site and cloud-based test environments is also presented. Selection of the tool for automating the setup process will be made based on criteria presented. In a company like Commit the added value of setting up the test environment with the revised process is high.

CONTENTS

1 INTRODUCTION

Commit has many projects each year where test environments are in use. Setting up the test environments takes an excessive amount of time and resources. Environment setup includes the creation of the needed virtual machines, powering machines on and ensuring the connections towards client systems.

The aim of this thesis is to find a quick, easy and cost-efficient way to start up a new virtual test-environment for the needs of a specific software development project. A secondary goal is to provide documentation for the test environment setup process.

The Research was conducted as action research to diagnose problems and weaknesses and help the researcher to develop practical solutions to the setup process and to address them quickly and efficiently.

The data used for this research was collected during the summer and the fall of 2018. The data used for this research was collected while doing setups of test environments.

Results conducted in this study can be applied to other small and midsized companies if the starting points for starting up and managing the test environments are the same as Commit has. However, using the results for other companies should be done with great care.

Another aspect of the test environments efficiency is the cost of the hardware used for the environments. Two different approaches, on-site and cloud-based environments will be investigated and possible cost differences between the two approaches are presented.

The company's projects are mainly fast paced, and the test environments need to be changed frequently. Therefore, an efficient process for setting up the test environments is needed. In this research such a process will be developed. The plan is to use the process in future work at Commit.

In the thesis summary the gathered information will be analyzed and further developmens for setting up the test environment will be discussed.

1.1 Commit

This master's thesis was made in co-operation with Commit, a midsized information technology company located in Espoo, Finland. Commit develops and sells software solutions for healthcare and other organizations. The software includes products for workflow, resource and quality management within radiological or surgical departments or any screening or follow-up programme. Commit also distributes off-the-shelf products from selected partners to supplement its own solutions and expertise.

Since 1989 Commit has implemented hundreds of projects where the application or integration environment has been built into a solution according to the customer's requirements. The solutions have linked together systems, business processes and people in an efficient, business-enhancing manner (Commit, 2018).

A multiprofessional advisory group was created in the company in order to provide different perspectives and inputs for the thesis. The advisory group included a developer, an IT-infrastructure specialist and a researcher.

1.2 Purpose and Objectives

The aim of this thesis is to find a quick, easy and cost-efficient way to start up a new virtual test-environment for the needs of a specific software development project. Secondary goal is to provide documentation for test environment setup process.

CHALLENGES
Currently there is no real documentation available for the test environment setup process. The goal is to fix this deficiency during the thesis work.

Another aim of this thesis is to compare expenses between on-site test environments and cloud-based test environments. Is it wise for a small or midsized company to build

their own on-site test environment or should they rent a cloud-based test environment when they need it?

IMPORTANCE OF THE THESIS

- Relevance to the test-environment automation
  - will help to allocate more time for actual testing and less for setup,
  - save time in general in projects,
  - save money for Commit,
  - will make connections easier between different test environments,
  - enable Commit personnel to start up test environments without the help of IT-support.
- Relevance to researcher
  - gain more knowledge,
  - help to understand the setup process of test environments,
  - help to understand the test process overall.

The study will look at opportunities for Commit to allocate more time and money for actual testing of software projects and less for setup of test-environments.

1.3 Research structure

Chapter 1 consists of a brief introduction of Commit and explains the background of the research.

Chapter 2 contains a literature review from different perspectives to understand the benefits of virtualizing the test environment. It gives an insight of the differences between on-site-environments and cloud-based test environments.

Chapter 3 will give an overview of how the setup processes are currently implemented in the company and shows general challenges with test environments. In chapter 4 methodology and research process about the setup will be addressed.
Action research together with a quantitative method was used in this study. The advisory group was launched at an early stage of the study.

Chapter 5 contains the data collection, chapter 6 gives information about operating with Ansible and chapter 7 goes through the results of the research and will give suggestions for future development.

2 LITERATURE REVIEW

This chapter is divided into multiple subtitles. In the following chapters I present prevailing concepts and theories that are relevant for the study. The research covers topics like automating the setup of the test environment and capturing and processing the received data.

Next, the basic information of test environments will be introduced. Why do companies need test environments, how they can be managed and launched, and what is the positive outcome from the testing? At the end of the section benefits of the automatized setup will be presented.

2.1 Why do we need test environments?

A test environment is a setup of software and hardware on which the test team is going to perform the testing (Guru99, 2019). Usually an environment is using servers, firewalls, switches, storage systems and operating systems.

A test environment is needed to be able to find system bugs in the software which is tested. Test environments can be diverse e.g. pre-production environment or the environment for integration testing. There can be different types of testing environments. Some of the test environments resembles production environment as closely as possible and some are simplified versions of the production environment. Simplified versions of the environments are used mainly to be able to start the testing phase faster (Khanduja, what is a testing environment for software testing, 2018).

Some of the projects are using more than one environment for testing. In general software-projects are fast paced, which means that project needs to adjust quickly to changing infrastructural situations.

IT-environments and software are often complicated to use and handle. Software that needs to be deployed might have several versions. After changing one part of the software there is still a need to run tests for the whole software since the changes might affect the entire software.

It is possible to end up in a situation where any other environment than the production environment is considered as a low priority environment. If companies are working like this, it is possible that managing, configuring and starting up test environments is considered to only spend time, money and other resources in the project. In the worst-case, not maintaining the test environments properly can lead to delays in the production release date.

Test environments can also be fully cloud based. In this case the company does not own used equipment and needed licenses. In small and midsized companies, the expenses of maintaining the environments might become too high. It is possible to cut costs which are associated with the test environments by using fully cloud-based solutions.

2.2 Cloud-based test environment versus on-premise hardware

Small and midsized companies can make decisions regarding the hardware used in test environments. Environments can be on-premise environment or fully cloud-based. There are many services available for using cloud-based test environments.

The on-premise test environment which Commit is using consists servers, firewalls, switches, SAN-switches and a storage system. On top of all of this there are virtualization software and operating systems. The cost of an on-premise test environment consists of the cost of hardware and the cost of software. Additionally, major cost factor is the licenses needed for the environment systems, systems like VMware, operating system and storage systems. The total cost of the environment was nearly 500 000€ when it was purchased.

Many cloud-based systems have several different options and pricing levels to use. Often a company would only pay for what they use without any long-term commitment. There are two components that affects to pricing:

1. an hourly per instance usage fee and,

2. a dedicated per region fee (AWS, 2019).

In a typical project at Commit the test environment is up and running approximately for 720h which means one-month usage for the environment. The actual testing time is shorter. Current test environments are mainly using two virtual CPUs, four gigabytes memory, 30 gigabytes storage and an operating system with needed licenses.

If Commit would have been starting to use cloud-based environments the pricing could have looked approximately like this:

- 2 US Dollars per hour for region fee,
- 0,237 US Dollars per hour for a one dedicated environment.

The list price for a 720h project would be 1610,64 US Dollars per month. Prices are from Amazon Web Services. AWS is a secure cloud services platform, offering computing power, database storage, content delivery and other functionality to help business scale and grow (AWS, 2019).

Based on the prices shown above small or midsized companies would be able to use cloud-based test environments roughly in 310 projects with the price of the on-premise hardware and licenses.

Usually there are less than 10 projects in a year, which means that company needs to use the same hardware for 31 years to make it profitable. However, normally the hardware needs to be updated after approximately five years of use.

2.3 Test process

Software testing is a process or a method of finding errors in a software application. The process is purposely developed to identify errors and missing instances where the system does not fulfill the requirements. With successful testing there is a reduced need of hot-fixes and maintenance when the software is in production (Immonen, Johdatus ohjelmistotuotantoon, 2018).

The testing done during a typical project can be of several different types, such as:

- System integration testing,
- user acceptance testing,
- performance testing,
- security testing and
- functional testing.

System integration testing is a type of testing where a testing team is performing tests, which try to verify the whole functionality of the software. User acceptance testing is generally performed by business and end users of the application or the software.

Performance and security tests are usually done simultaneously. These tests are non-functional and mainly performed by a specialist.

Testing includes tasks such as defining test cases to the test plan, setup of the test environment, running the test cases and analyzing and reporting the results of testing. It is wise to do very precise test case design. Exact and well-designed test cases give the possibility to find as many errors from the software as possible (Immonen, Johdatus ohjelmistotuotantoon, 2018).

Data received during the tests can be used in automated functional tests. Automated functional tests will give the opportunity to find errors from the software-code when the software is further developed. However, occasionally after the project has ended the data created during the test period is not saved, either because the data is not considered useful, or because saving the data is simply forgotten.

The test cases are made to confirm that the desired functionality will be tested during a project. A Test scenario is defined as any functionality that can be tested (Guru99, 2019).

After all the developing is done for the project, an installation package will be created. This packet is installed to the test environment to check if the software can be correctly

installed to the production environment. It is done as a last phase before the end user has first interaction with the end product (ProfessionalQA.com, 2019) and before software is released to production.

All types of testing have to be planned and after running all the test cases the results need to be analyzed. Analysis is needed to be able to inform the project group about the test results in various details and to provide evidence that the testing output meets the design input specifications (Guru99, 2019). Testing phases need to be planned and scheduled. In the project planning phase, some extra time is allocated to be used in case of emergencies. Still, often setup and maintenance of the environment takes more time than planned and this time is usually taken from the scheduled testing time.

Before the launch of an environment and the testing phase, objectives and goals for the actual testing, should be defined. What do we want to test and what do we want to achieve with the testing? What kind of test cases do we want to use? Which version of the software we need to use while testing?

2.4 Link between test process and test environment

Without an environment no actual application or software testing can be run and since testing changes directly in production where users or customers actively use the system is risky. (Ellison, software testing best practices, 2019) we understand the need for a test system.

During test environment setup it is a good opportunity to make environment-specific choices, which will help to run specific test cases in an environment.
It is a good idea to have a meeting between infrastructure-team, testers and developers to be able to setup a correctly specified environment which is needed during the test period.

Before testing can be started, the environment needs to be configured as per need of the application or software under test. Setting up a right test environment gives a change for software testing success (Guru99, 2019). Once the environment is ready to

be used, the data used in testing needs to be copied to the system, or there needs to be a way to produce it.

In some rare cases the data can be copied straight from the production environment. If production data can be used automatically, it will help to find errors in the code which is used in the production environment. In most cases the project will produce new software that differs from the production software. This means that it is not possible to use data straight from the production environment in the testing phase.

2.5 Benefits of automated test environment setup

Starting from a fresh OS installation, manually installing and configuring a full development environment can take several days. Without any automation it is easy to forget to install or configure environment correctly (Tharpe, Automatic development environment setup, 2019). Usually there are many kinds of software and hardware installations which need to be done in order to use environment for testing purposes.

There are many possibilities to make the process of setting up a complete test environment easier (Pronschinske, four methods to automate development environment setup, 2019). We could use installation tools like Ansible or disk images to install pre-configured hosts to test environment.

By automating the setup of the test environments, we are able to achieve several benefits. For example, when the number of necessary software licenses is known in advance and there is no need to change them at the end of the project, we can save time and money. With automated configuration of environments, the development and test environments are easier to keep consistent with production environment (Cloudbilimited, 2019).

Beside the licenses mentioned above, environmental settings made during the setup can be automated. After automating the setup and shutdown of test environments we can reduce error-prone and time-consuming manual steps. These include for example,

starting of the virtual machines, installing IP addresses and installing a dedicated program version on a virtual machine.

All the above-mentioned means that fewer mistakes will be made during environment setup. In other words, resources needed for the project can be used in software development which would save the project's resources – time and money.

As a summary, testing, management and the setup of the test environments are good to plan, define and document before the project starts. Good preparations provide a good start for the whole project and will save resources - time and money.

3 TEST ENVIRONMENT SETUP AUTOMATION

After having presented the literature review in the earlier chapters, this chapter gives an outline of action research method, an outline of automating the setup of a virtual test environment and an overview of how the setup is currently done.

While examining the existing material it was difficult to find any research carried out in which the setup, maintenance and economic issues of the test environments where considered from the point of view of a small and midsized company. This study tries to fill this gap and pave the way for more similar research.

Why do we start the test environments like we do? How can we do it easier? What changes needs to be done to the process?

3.1 Action research

As a research method for this study action research was chosen, because it provides the opportunity for a researcher to take part in the study and improve existing process together with an advisory group. It is a qualitative research methodology and according to Stephen Corey this participatory research is the process by which practitioners attempt to study their problems scientifically in order to guide, correct and evaluate their decisions and actions (Action research, 2018).

A typical action research process is cyclical and there are four different steps, planning, acting, observing and reflecting. The figure below presents a normal use of the action research method.

Figure 1 – Action research - (https://edresearch.nmsu.edu/research/cro-research-publications-2/action-research-initiatives/ar/)

Often a research starts when there is a dilemma that needs to be studied. The first step is to examine the problem carefully while gathering more facts about the situation into a *preliminary overall plan*.

In this thesis the goal is to improve existing process and study which changes should be done for the setup process in order to make it more efficient.

After a preliminary diagnosis is done more detailed *planning phase* is started. Here a detailed research plan is made. The plan describes of the actions, which will be changed in the setup process. The plan also describes how the changes will be carried out and how to observe whether the improvements are affective.

Next comes the *action phase*, where the research plan is carried out. During this phase minor changes to the plan can be made based on experience and feedback. In this phase new insights are likely to arise.

After the acting phase comes the *Observation phase,* which gives possibility to monitor changes made to the process and helps to assess the effect of actions. Every member

of the action research should keep a journal where observations are recorded for the use of the reflection phase.

In a *reflection* phase, results are critically reflected based on an observation and if needed, a new plan for acting phase is done (Stages of action research, 2019).

3.2 The test environment system at Commit

The test environment system at Commit consists around 40 different virtualized test environments in two physical servers. The demand to assemble test-environments comes from project needs. Many of Commit's projects are fast-paced. In the light of this, setting up a test environment and running the actual tests is wanted to be done in a fast and uncomplicated way. The time used to start the test environment is taken from the time reserved for developing or testing the new software.

In a usual project where there is a need to setup a test environment, two virtualized servers are started up and the newest version of the Commit software is installed on top of a regular Linux operating system. Besides Linux, Java, python, unzip and jetty are installed. Some of the projects requires more virtualized servers. On a project like this, seven virtualized servers are installed with Linux and other software as mentioned above.

Commit has many projects each year where test environments need to be used together with the client systems. Test environments used in projects are different and they are testing various things. Test data used during the project needs to be saved for future use after the project ends.

There are some challenges when creating the test environment as well as when storing the data after the project ends. Especially the setup process has to be revised since that phase of the project tends to take a lot of time.

Currently all the virtual machines are left powered on after the project ends, however, the aim is to shut them down when a project end. The virtual serves are not deleted,

and the project material is not always stored for the use of other projects. This part of the project needs to be planned better.

In addition to software installations and ensuring data resources, network connections towards the client systems may change as projects are changing. This might lead to changes in firewall and other network settings.

3.2.1 Observing the first start up

The following is an example of how the virtual test environment is reserved and launched at Commit. At the end I will present the time and the number of mouse clicks used for preparing the test environment for the use of a specific project.

After a new project has started, the environment that has been started up is linked together with the client-side. A test environment allocation is made in a shared document where ongoing project name, contact person's name and ending date of the project is stored. In some cases, the environment is not free for use for the newly started project because some other project is still using it. In a situation like this there is a need to wait until the environment is free to use or a new environment has to be selected for the project. If the environment is free, there is still a need to ensure that the data stored to the environment, from a previous project, is backed up before deleting it from the environment.

A test environment is created in VMware by using a pre-installed template. The name and location of the virtual machine and an attached storage system will be assigned manually during the setup of the environment

After all the detailed information has been entered into the new environment, the virtual machine can be powered on.

When installation is finished and the operating system has installed into virtual machine, the system will be prompt for a NetBIOS name and IP-address. An IP-address is not assigned automatically, instead it needs to be manually looked up from a list of

pre-selected IP-addresses. This phase of the test environment setup took 20 minutes and used 28 mouse clicks.

Later when NetBIOS name and IP have been given to the system, the virtual machine initialization begins. Initialization phase takes about five minutes and after that it is possible to log in to the system with a pre-defined username and password. The virtual machine will be registered to the company's network and all the new system updates need to be installed to the system. This operation can take up to fifteen minutes.

Immediately after all the updates have been installed the specific project installation can be made. Installation covers version of the software which will be used during the testing phase and also configurations for communicating with third-party systems are done. Software installations and configurations towards third-party systems took 17 minutes and 15 mouse clicks.

In the best possible situation, a test environment for the need of a project can be built using VMware with 20 mouse clicks. Those clicks don't include any additional software installations which might be needed on the user's own computer or clicks needed when searching the IP-addresses to be used in the initialized environment.

The example initialization of the requested environment took 57 minutes, and 43 mouse clicks were needed. Before testing will be started, the environment needs to be configured and needed software to be installed as per need of the application or software under test. Without any automation it is possible that we forget to install or configure the environment in a needed way. Installation tools could make the process simpler and less prone to errors.

3.3 Overall plan at Commit

After the first setup, it was clear that the setup and management of test environments is taking too much time and the need for an automated setup was clear. Therefore, the setup process needed to be changed.

As a researcher I started to think if it would be possible to manage environments in a more user-friendly way? Would I be able to start a new environment with a single script? Would I be able to collect all the data from the test environment with a single script after all testing is done?

Since Commit has around 40 different test environments, and it took 57 minutes time and 43 mouse clicks to setup a single new test environment, a need to change the setup process was clear.

The advisory group suggested that it would be beneficial to use some simple IT-automation or installation tool while doing the setup.

If we choose to use software for automated test environment setup, we would gain real benefits if the software could:

- setup the environment with a script.
- While setup, install newest updates into operating system.
- Install operating system with project defaults (users, system names, IP's).
- Install the software to be tested during the project

4 RESEARCH METHODOLOGY

4.1 Research design

Research design is defined as a framework of methods and techniques chosen by a researcher to combine various components of research in a reasonably logical manner so that the research problem is efficiently handled. It provides insights about how to conduct research using a particular methodology (Bhat, 2018). This study will be done as a collaborative action research, where more than one person is involved in the implementation of a new process.

4.2 Research strategy

In this study I am aiming to improve an existing process of managing and automating the setup of test environment systems. A quantitative approach will be used in this study. Quantitative methods are research methods dealing with numbers and anything that is measurable in a systematic way of investigation of phenomena and their relationships. It is used to answer questions on relationships within measurable variables with an intention to explain, predict and control a phenomenon (Quantitative Research methodology, 2019).

The aim for the researcher is to study the existing process in detail, propose improvements for the process to an advisory group and in general gain more knowledge about the subject and the process.

In the field of ICT, quantitative methods often deal with results computation and system analysis using a scientific approach (Quantitative Research methodology, 2019). After the existing process has been improved there should be a possibility to see positive measurable progress in the process of starting up the test environments.

5 DATA COLLECTION

Data collection is the process of gathering and measuring information on variables of interest, in an established systematic fashion that enables one to answer stated research questions, test hypotheses, and evaluate outcomes (Data collection, 2019). After the first setup, it was clear that the setup and management is taking too much time. Therefore, the process needed to be revised.

This chapter begins with presenting primary and secondary data collection methods. Afterwards a more detailed plan for building the desired test environments is presented and at the end the outcome will be presented.

5.1 Data collection method

Data collection is a process of collecting information from all the relevant sources to find answers to the research problem. Data collection method can be divided into two categories: secondary methods of data collection and primary methods of data collection (Data collection, 2019).

5.1.1 Primary data

Primary data collection methods can be divided into two groups: quantitative and qualitative (Data collection, 2019). This study is based on primary data with quantitative approach. Data was collected with two different projects in which there was a need to start up a test environment.

The data collection was mainly done by observing the setup, interviewing the advisory group and building the test environments. The meetings with the advisory group aimed to understand which steps were most problematic during the process of starting up the needed environments.

5.1.2 Secondary data

Secondary data is a type of data that has already been published in books, newspapers, articles, magazines, journals, online portals and so on (Data collection, 2019). The secondary data used in this study is mainly collected from online portals, articles and books.

5.2 Detailed plan

If you fail to plan, you plan to fail (Goodreads,2019). The action plan summarizes the steps I will follow to be able to answer the research question.

In the table below is the action plan for improving the process of starting up the test environments.

Timeline	**Actions – What will we do?**	**Desired outputs**	**What data to collect?**	**How we will interpret what the data means?**	**Who is responsible?**	**etc.**
Summer 2018	Discuss with advisory group about challenges in the setup process	An overview of challenges and gain more detailed info about what to study	Advisory groups review of the situation	Prepare memo about situation, recognize phases to improve	Researcher / Advisory group	
Summer 2018	Prioritize improvements to the setup process based on discussion	More detailed info about desired changes			Researcher / Advisory group	
Summer 2018	Study IT automation and installation tools	Seek different possibilities.	Differences in automation/installation tools		Researcher / Advisory group	
Fall 2018	Choose which tool to be used	Choose the best solution for Commit			Researcher	
Fall 2018	Operate newly selected automation tool	More effective process	How to use the selected tool		Researcher	
Fall 2018	Design improvements to the process	A new process	Revised process with setup scripts		Researcher	

	with the help of the automation tool					
Fall 2018	Take the revised process into use				Researcher	
Fall 2018	Observe changes	Measure changes in the process	Differences between old and new process. Measured setup times and number of mouse clicks		Researcher	
Spring 2019	Analyze and discuss results from the new process	Discussion with advisory group based on the new start up process	Advisory groups review of the situation		Researcher / Advisory group	
Spring 2019	Report results	Create final report	Thesis report		Researcher	

The action plan makes it clear what needs to be done and by who in order to achieve an objective. From the action plan all members of the research can see what the goal of the study is. It will help the researcher to keep track of issues that needs to be taken care of and helps the researcher when it is time to do the reporting. The action plan is a working document which can be modified at any time if the researcher finds a new course to a more effective end result.

5.3 Installation tool selection

The advisory group suggested that it would be beneficial to use some kind of IT-automation or installation tool while doing the environment setup. There is a multitude of different software available on the market and users should know what their needs are, what each software can offer and which software best fits to their requirements.

As a researcher I started by making the selection and evaluation criteria for Commit. The selected software should meet the following criteria:

- Free to use for small or middle-sized companies,
- no need for client-installation,
- support VMware installation,
- automate configuration management,
- support for scripts,
- easy to operate,
- documented and supported by developers.

The software selection was made by the researcher on the basis of the selection criteria.

6 ANSIBLE

Ansible is a radically simple IT automation engine that automates cloud provisioning, configuration management, application deployment, infrastructure-service orchestration and many other IT needs (Ansible, 2019).

Ansible can be used freely in a small company. No actual licenses are needed.
It will give the possibility to save time and be more productive, eliminate errors and repetitive tasks. Ansible includes many modules to support a wide variety of integrations, including VMware (Ansible, 2019). which is used by Commit.

6.1 Operating with Ansible

Operating with Ansible starts with playbook creations. Playbooks are Ansible's configuration, deployment and orchestration language, which executes selected commands in wanted order. Playbook can describe a policy you want your remote systems to enforce, or a set of steps in a general IT process (Ansible playbooks, 2019). Ansible starts up the environments according to a playbook. Playbook contains plays which are seen as a task in ansible. The selected playbook can be started from a command line and after starting Ansible will run all the defined tasks from the selected playbook.

Playbooks are expressed in YAML format and have a minimum of syntax, which intentionally tries not to be a programming language or script, but rather a model of a configuration process (Ansible playbooks, 2019). One playbook can contain a single or multiple play which are run in the order specified in the playbook.

After the playbooks are created, it is wise to create host-file. The host-file contains all the relevant information for the needed environment. Information such as:

- The environment name,
- host name,
- included virtual machines for the environment,
- VMware template used to start the wanted virtual machine,

- Selected storage system,
- the IP-addresses to be used.

When the host file is created and all the information is updated to the file, tasks used for the setup and deleting the environments can be used. For each play in a playbook, you get to choose which virtual machines you're targeting, and which username will be used when logging into the remote servers. (Ansible playbooks, 2019) If there are any changes to the environments only the host-file needs to be updated. At the moment all the work with the host file is done manually.

The host file can contain information about environments which are not similar. In some of the environments there can be more than one virtual machine. The following example contains configuration for two different kinds of environments:

```
[FullTestEnv-1]
VirtualMachine1-t1 custom_ip=101.202.100.84
VirtualMachine2-t1 custom_ip=101.202.100.85
VirtualMachine3-t1 custom_ip=101.202.100.86
VirtualMachine4-t1 custom_ip=101.202.100.87
VirtualMachine5-t1 custom_ip=101.202.100.88
VirtualMachine6-t1 custom_ip=101.202.100.89
VirtualMachine7-t1 custom_ip=101.202.100.90
VirtualMachine8-t1 custom_ip=101.202.100.91

[FullTestEnv-1: vars]
folder=Company/E1

[SmallTestEnv-s]
TestEnv-11 custom_ip=100.101.202.211
TestEnv-12 custom_ip=100.101.202.212

[SmallTestEnv-s:vars]
folder=/Company
```

In a rare case where any IP or name would be changed on the company's infrastructure, the only place where the change is needed is the host file.

6.2 Starting and deleting environment with Ansible

When starting the environment only one playbook needs to be run. The play will ask a description of the environment created and will launch the specific environment, install all needed packages and update them all to the latest version.

The play starts with the deploy -command.

```
---
- hosts: all
##- hosts: "{{vmname}} ‟
  connection: local

  vars_prompt:
    - name: "notes"
      prompt: "Kuvaus"
      private: no
      default: ""
#      name: "vmname"
#      prompt: "vmname"
  tasks:

  - set_fact: creationdate="{{lookup ('pipe','date
"+%Y/%m/%d %H: %M"')}}"

  - name: Deploy vm
    vmware_guest:
      hostname: vcsa-t.company.int
      username: company@vsphere.company
      password: C0mpany!
      validate_certs: False
      datacenter: "Datecenter-Company-Test"
      cluster: "Company_Test"
      folder: "{{folder}} ‟
      name: "{{inventory_hostname}} ‟
      template: Company-template
      networks:
      - name: Portgroup_Testiverkko
        ip: "{{custom_ip}} ‟
        netmask: 255.255.255.255
        gateway: 100.202.100.1
        domain: ocvk.int
        dns_servers:
        - 1.2.3.4
        - 1.2.3.5
        type: static
      customization:
        dns_servers:
        - 100.2.255.1
        - 100.2.255.2
        domain: Company.test
      state: poweredon
      wait_for_ip_address: yes
```

```
      annotation: "{{notes}} - Creation date: {{crea-
tiondate}} "
    register: deploy
    tags:
       - deploy
    delegate_to: localhost


- import_playbook: update_guest.yml2
- import_playbook: install_packages.yml

install_packages.yml
---
- hosts: all

  tasks:

  - name: ensure a list of packages installed
    yum:
      name: "{{packages }}"
    vars:
      packages:
      - notepad
      - photoshop
      - zip
      - unzip
      - Java
```

After the environment is no longer needed and it can be deleted, the delete playbook is needed to run in order to erase the desired environment.

```
hosts: all
  connection: local

  tasks:
  - name: Delete vm
    vmware_guest:
      hostname: vcsa-t.company.int
      username: company@vsphere.company
      password: C0mpany!
      validate_certs: False
      datacenter: "DC-T"
      cluster: "Cluster_Test"
      folder: "{{folder}} "
      name: "{{inventory_hostname}} "
      force: yes
      state: absent
    delegate_to: localhost
    tags:
       - delete
```

6.3 Acting for the second setup

The following is an example of how the virtual test environment, *FullTestEnv-1*, is reserved and launched at Commit with the newly revised process. At the end I will present the time and the number of mouse clicks used for preparing the test environment for the use of a specific project.

When the second project has started, the allocation is made on a shared document where ongoing project name, contact person's name and ending date are stored manually. In this case, the *FullTestEnv-1* is not available to use for the second project because some other project is still using it. I needed to find out if all the data from the environment is backed up before deleting it from the environment.

A new test environment is now started up with the help of a playbook created with the Ansible automation tool. The name and location of the virtual machine and an attached storage system will be assigned automatically during the setup of the environment.

The setup begins with logging into a control machine from where I can use Ansible automation tool through a terminal window. This phase took 4 mouse clicks. From terminal window I can launch the setup of the FullTestEnv-1, which contains eight different virtual machines. Start is done with the following command:

```
Ansible-playbook -v deploy_quest.yml2 –
limit=FullTestEnv-1
```

When a play is started, a script will ask for a description of the environment. In this case the project name is *mansikka.* As a first task the play will clone wanted virtual machines from a pre-installed template created with VMware.

When cloning of the virtual machines is ready, the play will continue by assigning a data storage system, IP-address and a host name to the newly started system. After this phase is done, the virtual machines will be registered to the company's network.

The host file contains information about software packages which have to be installed on top of the pre-created virtual machine. The next phase in the play is to install the desired software packages and update those to the newest version.

After all the software packages have been updated, the Ansible play will reboot all of the virtual machines. The reboot takes two minutes. When all the virtual machines are started and before the play ends, the play will ensure that all the packages mentioned in the package list are installed on the virtual machines.

Using Ansible for the setup of the needed virtual machines, installing operating system, host name and IP-address, connecting to storage system, upgrading selected software's and registering virtual machines to company's network took 24 minutes and eight mouse clicks.

After the play has finished and the new environment with all the needed virtual machines are reachable again, I can start the software installation for the specific project. The installation covers the version of the software which will be used during the testing phase. Configurations towards the client system are also made. Software installation and configurations towards the client systems took 12 minutes and 12 mouse clicks. The second initialization of the requested environment with revised process and Ansible automation tool took 36 minutes and 17 mouse clicks.

6.4 Results of the study

During the thesis there were two different projects which were selected to be pilot projects in order to study possibilities to start test environments in a more efficient way.

The allocation of the needed environment was done similarly in both projects, by manually entering the data to a shared document where all the relevant information regarding the project and environments are stored.

Both projects needed full test environments, all together nine virtualized servers, a storage system and Commit's software installed. First setup was done with an old process to be able to find challenging phases when starting up the test environment. First setup of the requested environment took 57 minutes and 43 mouse clicks.

Before the second setup was made the process was revised, and the Ansible IT-automation tool was used in order to make the setup more efficient. Before using the Ansible playbook there was a need to configure host-file and wanted plays which automate the setup process.

The second setup was made by following the new process and with the help of the new automation tool, Ansible which helped to overcome some challenges in the setup process. Second setup of the requested environment took 36 minutes, and 17 mouse clicks.

After the environments were started up, it was clear that the new process is efficient when compared to the old one.
The time used and the number of mouse clicks was measured when starting up the specific environment.

Process	**Environment**	**Number of started servers**	**Storage system used**	**Software installed**	**Time used (min)**	**Mouse clicks needed**
Old	Virtualized	9	X	Installed manually	57	43
New	Virtualized	9	X	Installed Manually	36	17

7 CONCLUSIONS

This section will present the conclusions of this master thesis where I sum up the usage of action research methodology, the benefits of the revised setup process and my recommendations for future actions in order to make the process even more efficient. The aim of the thesis was to create a more efficient test environment setup process for a medium-sized company.

The conclusions and recommendations shown in this thesis are based on the researcher's opinion and other factors in the company's interest. The research idea was clear already before the thesis work started and also the main phases where improvements were needed were known, because I have had a chance to setup environments before the actual study was even started. Discussions with the advisory group reaffirmed the phases which needed improvements.

The usage of action research for a study like this, where the aim is to improve existing processes was an ideal choice. During the first setup it was easy to observe and make notes about challenges in the setup process. Afterwards, when discussing about solution proposals with the advisory group the acting part became easy.

Ansible playbook examples and best practices can be downloaded from the home page and after having reviewed instructions operating with Ansible was natural and customizing the plays was relatively easy. When all the desired plays were customized for Commit's use it was easier and more efficient to start up new test environments compared to the old process. The use of Ansible allows personnel to concentrate on other tasks while Ansible is starting up the environment.
Using Ansible in a small company will not add any extra expenses for the IT-budget. Some costs arise from worktime used to study how to operate Ansible. In a company like Commit the added value of starting the test environment in a cost-effective way, e.g. with Ansible, is high.

7.1 Future actions

The first recommendation for future plans would be to automate the process of reserving the environments. Most of the mouse clicks needed when starting the environment came from manually selecting a shared document where information about the project was stored. The setup process is now automated with Ansible, and when creating a new environment, the play is asking for a description of the environment. It should be possible to ask for a detailed project name, a contact person's name and the end date and store this information automatically to a pre-selected destination, from where all the employees could find the information.

The second recommendation for future plans would be to implement an automatic installation, possibly using Ansible, of the software to be tested. This implementation can be done, but it needs to be under critical consideration how important and wise it is. This part would save more time and mouse click when setting up up a new test environment for the use of a specific project. The down side when trying to install software as part of the startup process is the that the software that should be installed is not necessarily ready at the same time as the setup of the environment is done.

The third and last recommendation is for deleting the test environment. When an environment is started up, and if an end date is entered through Ansible, it should be possible automatically run the delete environment play when end date (+ some extra safety margin) is reached.

It should be possible to the delete play so that all the relevant data (directories) would first be copied to a pre-selected data storage. This would help in storing the relevant data to be used in coming projects.

REFERENCES

Commit Oy. Retrieved May 23, 2018 from http://www.commit.fi/eng/company/

Action research - What is action research? Retrieved April 20, 2018 from (https://edresearch.nmsu.edu/research/cro-research-publications-2/action-research-initiatives/ar/)

Jaideep Khanduja – What is a testing environment for software testing – Retrieved Nov 19, 2018 from: https://itknowledgeexchange.techtarget.com/quality-assurance/what-is-a-testing-environment-for-software-testing/

Jarkko Immonen – Johdatus ohjelmistotuotantoon – Joensuun yliopisto (2002). Retrieved June 16, 2018 from http://cs.joensuu.fi/~jimmonen/jot_moniste/jot_moniste_121.html

Ascd.org – action research process. Retrieved April 22, 2018 from http://www.ascd.org/publications/books/100047/chapters/what-is-action-research¢.aspx

Action research article. Retrieved Nov 20, 2018 from: http://www.wou.edu/~girodm/library/ch9.pdf

Adi Bhat - Research design article. Retrieved Nov 21, 2018 from: https://www.questionpro.com/blog/research-design/

Data Collection Methods – Research methodology. Retrieved Nov 24, 2018 from: https://research-methodology.net/research-methods/data-collection/

Ansible IT automation. Retrieved Nov 28, 2018 from: https://www.ansible.com/

Working with playbooks. Retrieved Feb 11, 2019 from: https://docs.ansible.com/ansible/latest/user_guide/playbooks.html

What is Amazon web services? Retrieved Jan 09, 2019 from: https://aws.amazon.com/what-is-aws/

AWS Dedicated instances. Retrieved Dec 20, 2018 from: https://aws.amazon.com/ec2/purchasing-options/dedicated-instances/

What is test scenario? Template with examples. Retrieved Jan 09, 2019 from https://www.guru99.com/test-scenario.html

Installation testing. Retrieved Jan 14, 2019 from: http://www.professionalqa.com/installation-testing

Design Verification & Validation process. Retrieved Jan 14, 2019 from: https://www.guru99.com/design-verification-process.html#2

Richard Ellison, Software testing environments best practices. Retrieved Jan 16, 2019 from: http://www.softwaretestingmagazine.com/knowledge/software-testing-environments-best-practices/

James Tharpe, Automatic development environment setup. Retrieved Jan 16, 2019 from: https://www.jamestharpe.com/automatic-development-environment-setup/

Mitch Pronschinske, Four Methods to automate development environment setup. Retrieved Jan 16, 2019 from: https://dzone.com/articles/4-methods-automate-development

The Cloud bi limited integrated optimized approach to automation. Retrieved Jan 16, 2019 from: https://cloudbilimited.com/automated-environment-creation-and-configuration/

Quantitative Methods – University of Southern California. Retrieved Jan 16, 2019 from: http://libguides.usc.edu/writingguide/quantitative

What is action research. Retrieved Jan 28, 2019 from: https://www.sagepub.com/sites/default/files/upm-binaries/36584_01_Koshy_et_al_Ch_01.pdf

Stages of action research project. Retrieved Jan 29, 2019 from: http://cei.ust.hk/files/public/ar_intro_stages_of_an_action_research_project.pdf

Responsible Conduct in Data Management. Retrieved Feb 6, 2019 from: https://ori.hhs.gov/education/products/n_illinois_u/datamanagement/dctopic.html

Quantitative Research methods. Retrieved Feb 7, 2019 from: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=15&ved=2ahUKEwjfu__a1KngAhXGJSwKHU9jDO4QFjAOegQICRAC&url=https%3A%2F%2Fwww.tankonyvtar.hu%2Fhu%2Ftartalom%2Ftamop412A%2F2011-0021_22_research_methodology%2FCMRM6103_Research_methodology_08.pdf&usg=AOvVaw08oy99TDaODnpjbtfKtM2g

Goodreads, Planning quota. Retrieved Feb 7, 2019 from: https://www.goodreads.com/quotes/460142-if-you-fail-to-plan-you-are-planning-to-fail