jamk.fi

CI System Relocation

Jani Tuovila

Master's thesis
May 2016
Cyber Security
Master's Degree Programme in Information Technology

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

jamk.fi

Description

Author(s)	Type of publication	Date
Tuovila, Jani	Master's thesis	May 2016
		Language of publication: English
	66	Permission for web publication: (x)

Title of publication
CI System Relocation

Degree programme
Master's Degree Programme in Information Technology

Supervisor(s)
Salmikangas Esa, Huotari Jouni

Assigned by

Abstract

A network of partner companies has established a Virtual Enterprise in order to develop a complex telecommunication device for the customer. The project has been ongoing already for several years and during that time a complex CI system to support the R&D work has been built in the premises of the integrator. Now the customer wishes to gain control of the CI system, and the project planning for the relocation project is ongoing.

The author was hired as a consultant to work as a part of the relocation planning team. The primary goal was to solve security implications and other blocking impediments the relocation of the CI system might cause. A secondary goal was to meet the requirements placed by the customer. As a result, a suggestion on how to proceed with the relocation is provided to the customer.

The study was conducted as action research, where the author works as part of the planning team "intervening" during the meetings and discussions. Data triangulation was used when gathering information using interviews as well as documentation and literature reviews. GAP analysis was utilized, when analyzing the difference between the current state and the desired state.

The requirements given by the customer could not be met, however, a feasible way to proceed with relocation was found in the end.

Keywords/tags
Virtual Enterprise, Virtual Organization, Continuous Integration, Information System Relocation

Miscellaneous

jamk.fi

Kuvailulehti

Tekijä(t)	Julkaisun laji	Päivämäärä
Tuovila, Jani	Opinnäytetyö, ylempi AMK	Toukokuu 2016
		Julkaisun kieli: Englanti
	66	Verkkojulkaisulupa: (x)

Työn nimi
CI System Relocation

Koulutusohjelma
Informaatioteknologian koulutusohjelma

Työnohjaajat(t)
Salmikangas Esa, Huotari Jouni

Toimeksiantaja(t)
-

Tiivistelmä

Useiden yritysten partneriverkosto on perustanut virtuaaliyrityksen, jonka tavoitteena on ollut kehittää asiakkaalle monimutkainen tietoliikennelaite. Kehitysprojekti on ollut käynnissä jo useita vuosia, jonka aikana sitä tukemaan on rakennettu monimutkainen jatkuvan integraation (Continuos Integration) tietojärjestelmä. Tämä järjestelmä on ollut rakennettuna integroinnista vastaavan yrityksen tiloihin, mutta projekti sen siirtämiseksi asiakkaan tiloihin on jo käynnistetty.

Kirjoittaja työskenteli konsulttina järjestelmänsiirtoa suunnittelevassa tiimissä. Työn ensisijaisena tavoitteena oli selvittää siirrosta aiheutuvia tietoturvahaasteita sekä muita siirrosta aiheutuvia teknisiä esteitä. Toissijaisena tavoitteena oli täyttää asiakkaan asettamat vaatimukset siirrolle. Työn tuloksena asiakkaalle toimitettaisiin toimenpide-ehdotus, jonka mukaisesti muutossa tulisi edetä.

Tutkimus toteutettiin toimintatutkimuksena, jossa kirjoittaja toimi osana suunnittelutiimiä puuttuen tiimin työskentelyyn palavereissa ja muissa tapaamisissa. Tietoa kerättiin dokumentaatiota ja kirjallisuutta tutkimalla sekä projektihenkilöstöä haastattelemalla, kun taas nykytilanteen ja tavoitetilan välistä eroa analysoitiin puuteanalyysin avulla.

Tutkimuksessa ei päästy asiakkaan asettamien vaatimusten mukaiseen tavoitteeseen, mutta vaihtoehtoinen ratkaisuehdotus asiakkaalle saatiin kuitenkin toimitettua. Ratkaisuksi ehdotettiin tietojärjestelmän kopiointia aluksi sellaisenaan asiakkaan tiloihin, josta yksittäisiä palvelimia voitaisiin myöhemmin siirtää niille sopiviin paikkoihin.

Avainsanat (asiasanat)
Virtuaaliyritys, Virtuaaliorganisaatio, Jatkuva integraatio, Informaatiojärjestelmän uudelleen sijoittelu

Muut tiedot

Contents

1 Introduction 4
1.1 Background 4
1.2 Continuous Integration 5
1.3 Objectives 6
1.4 Research methods 7
1.4.1 Research Approach 7
1.4.2 Research Process 8
1.4.3 Data Collecting and Analysis Methods 10

2 Virtual Enterprise 12
2.1 Overview of Virtual Enterprise 12
2.2 Information Systems Interconnection 12
2.3 Securing the interconnection. 14
2.3.1 Leased Line 15
2.3.2 VPN 15
2.3.3 DMZ 16
2.3.4 Reverse Proxy 17
2.3.5 Desktop Virtualization 17

3 The Current State 20
3.1 Overview 20
3.2 Network Connections 23
3.2.1 VPN (gateway-to-gateway) 23
3.2.2 Leased line (Fiber) 24
3.3 Tools 25
3.3.1 Atlassian JIRA 25
3.3.2 SpiraTest 27
3.3.3 Jenkins 28
3.3.4 GIT/Gerrit/Gitorious 30
3.3.5 Data mart 32
3.3.6 Nagios 32
3.3.7 Email 33

3.3.8 Team site 33
3.3.9 Developer and Test Automation Tools 34
3.3.10 Backups 35
3.4 Use Cases 35
3.4.1 CI Machinery 35
3.4.2 Middleware Component CI 36
3.4.3 SW Component CI 39
3.4.4 Hardware 42

4 The Desired State 44
4.1 Overview 44
4.2 Network connections 45
4.2.1 Virtual Desktop (Citrix) 46
4.2.2 VPN (Host-to-gateway) 48
4.2.3 Leased line (fiber) 49
4.3 Tools 50
4.3.1 Atlassian JIRA 50
4.3.2 SpiraTest 51
4.3.3 Gerrit/GIT 52
4.3.4 Email 53
4.3.5 Teamsite 53
4.3.6 Backups 54

5 Summary 55

6 Conclusion 58

References 61

Figures

Figure 1. Action Research Protocol after Kemmis 9
Figure 2. A Virtual Organisation network example. 13
Figure 3. Interconnection Components 15
Figure 4. A Layered DMZ Implementation 17
Figure 5. High level picture of current situation 20
Figure 6. Initial Setup 22
Figure 7. JIRA access diagram 26
Figure 8. High level picture of CI Machinery 36
Figure 9. Middleware Component CI Use Case Diagram 37
Figure 10. SW Component CI Use case diagram 40
Figure 11. HW Use case diagram 43
Figure 12. High level picture of target system. 44
Figure 13. Initial plan for CI Infrastructure in the customer premises 45

Tables

Table 1. Jira Dependencies 27
Table 2. SpiraTest users and dependencies 28
Table 3. Jenkins connections 30
Table 4. Gerrit and Gitorious connections 31
Table 5. Datamart dependencies 32

1 Introduction

1.1 Background

A network of partner companies has established a Virtual Enterprise in order to develop a complex telecommunication device for the customer. The Virtual Enterprise including more than 10 companies (partners and subcontractors) in multiple sites around the world has been ongoing already for several years. During that time there have been several IT systems built to support the R&D work, which has caused the overall R&D environment to be quite complex. One company which has been operating as an integrator for the device has had the main responsibility for most of the IT systems (CI system) used for the R&D. As the customer has established a local site, the R&D systems are planned to be relocated to the customer premises. Because of the complexity of the CI system and different kind of network architecture and security policies between the companies, the relocation of the CI system was expected to be a very challenging task.

The planning of IT system relocation was already ongoing, when the author of this study was hired to the project as a consultant with a fixed contract. The original purpose was to ensure the security in the relocated R&D environment. It soon became clear that the planning was still in its infancy and evaluating the security measures would not be possible in the given time frame; therefore, the research problem later evolved to finding out a viable solution that would enable the relocation.

Because of the issues with confidentiality and publicity, the companies involved want to remain anonymous, which means the names of the companies and persons will not be published. Because of the confidentiality issues, any sensitive information about the information systems nor the projects of the companies will not be published either.

1.2 Continuous Integration

Continuous Integration is a software development practice where members of a team integrate their work frequently; usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible. Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly. (Fowler, 2006)

Continuous Integration (CI) is one of the core ideas behind the Agile Software Development. It decreases the risk of integration conflicts and allows a fast feedback loop for developers. It involves making small changes to software and then building and applying quality assurance process. CI forces the defects to emerge early, rather than waiting for software to be fully produced. If defects are caught in the later stages of the software development lifecycle, the process will be more expensive. The cost of repair radically increases as soon as the bugs escape to production. Estimates suggest it is 100 to 1000 times cheaper to capture defects early. (Berg, 2012, 1)

Quality assurance process applied after build involves different levels of automatic and manual testing. The code of developers can be smoke tested automatically when the code is committed to the version control system. In addition to automatic smoke testing, a peer review and an automatic static analysis can be deployed before the commit is accepted to the mainline. In the end of each day, a daily build is built including all the commits received during the day. Daily build involves more rigorous test automation campaign during the night time. In the end of each sprint (usually two weeks), a latest verified daily build is promoted to a release, which involves automated release and regression testing.

This kind of CI system demands build power, network capacity and test automation capacity. As each developer may commit several times a day and if there are several teams of developers there could be 50-200 builds per day. As each build needs to be transferred to test automation places, the load of the network infrastructure can be

quite substantial. Enough automated test places are needed to keep the feedback loop fast enough, as each build needs different level automatic tests (smoke, daily, release or regression tests). Each part of the machinery needs to be in order for developers to get their feedback fast and the mainline to remain in good shape. As Fowler states (2006), a key part of doing a continuous build is that if the mainline build fails, it needs to be fixed right away. A broken mainline could prevent developers from pushing commits which could cause lost work hours. In a large organization it could cost dearly.

1.3 Objectives

The original research problem was to find out the security implications the CI system relocation from partner to own premises might cause. This research problem later transformed not just to finding out and solving the security implications but other blocking impediments as well. The reason for this change was the fact that the planning was nowhere near ready, and evaluating the security implications was not yet feasible.

In addition to the research problem, the customer gave two requirements for services to be used from the internal service catalog. The first one was to use the private cloud the customer was building in parallel, as much as possible. The second requirement was to use a virtual desktop environment based on Citrix as the main working method for the partners.

> *"If you do not know where you come from, then you don't know where you are, and if you don't know where you are, then you don't know where you're going. And if you don't know where you're going, you're probably going wrong." (Terry Pratchett)*

To reach the target, the present situation needed to be clarified, which meant the current state including the use cases needed to be analyzed. Once the current state was known the target state could be defined and differences analyzed in order to get

the steps needed to reach the target. Thus the output of this study would be the steps needed to reach the objectives, and to get there following steps needed to be made:

1. What is the current CI system like?
2. Who are using the CI system?
3. How the CI system is used (use cases)?
4. What is the desired state?
5. How to reach the desired state?

This study focuses on technical perspective and even though the non-technical issues are reported to the customer, they will not be included in this study.

1.4 Research methods

1.4.1 Research Approach

According to Gilmore, Krantz, and Ramirez (1986, 160), Clark (1976) states the action research aims to contribute both to the practical concerns of people in an immediate problematic situation and to further the goals of social science simultaneously. Thus, there is a dual commitment in action research to study a system and concurrently to collaborate with the members of the system in changing it into what is together regarded as a desirable direction. Accomplishing this twin goal requires the active collaboration of researcher and client, and thus it stresses the importance of co-learning as a primary aspect of the research process. While action research sets out to make scientific contributions and solve practical problems within the context of distinct projects or groups of projects, the relative proportions of “intervention” and “research” may vary according to the circumstances.

As the author of this study was enlisted as a consultant for the CI relocation project and worked as part of the team planning and preparing the relocation, the action research as a research method was a natural choice. The author was not involved in

the technical work such as server installations but was attending the meetings, discussions and data gathering and “intervened” during these engagements.

What separates this type of research from general professional practices, consulting, or daily problem-solving is the emphasis on scientific study, which is to say the researcher studies the problem systematically and ensures the intervention is informed by theoretical considerations. Much of the researcher’s time is spent on refining the methodological tools to suit the exigencies of the situation, and on collecting, analyzing, and presenting data on an ongoing, cyclical basis. (O’Brien, 1998.)

1.4.2 Research Process

The study starts with a planning phase, where the goals, risks, requirements, schedule etc. are defined for the project. Also, the thesis scope is narrowed and exclusions are defined. After the planning phase the current state is analyzed. For this analysis, the data is gathered through interviews and documentation walkthrough, and related literature is reviewed for a more effective analysis.

Once the current state has been analyzed the desired state analysis follows, which is carried out by gathering the requirements outlined by the customer and the current state analysis. Once the desired state is clear, the current offering of the customer service catalog is analyzed. Afterwards the current state, desired state and the service catalog are analyzed for steps to reach the desired state.

According to O’Brien (1998), Kemmis has developed a simple model of the cyclical nature of the typical action research process (Figure 1). Each cycle has four steps: plan, act, observe, reflect.

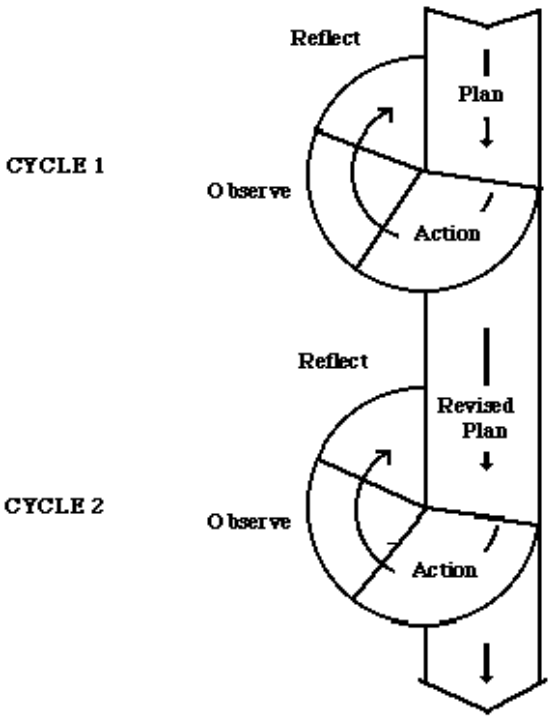

Figure 1. Action Research Protocol after Kemmis (MacIsaac, 1995)

The cyclic nature of the study is manifested in the phases starting from the current state analysis and ending with the analysis of the current state, desired state and the service catalog. Even though the majority of the current state and the desired state analysis are done in the first cycle, the subsequent cycles may further define these analyses. During each cycle a component from the existing CI system is taken under analysis, and a new location is sought for it based on the desired state and the service catalog. Each cycle does not necessary involve the technical part (the real "action"), where a technical resource is implemented. Instead on some cycles only the requirements led from the current state analysis, use cases and restrictions of the service customer is offering are analyzed and if there is a conflict observed, the technical phase (or "action") does not take place.

1.4.3 Data Collecting and Analysis Methods

In action research data comes through engagement with others in the action research cycles. Therefore, it is important to know that acts intended to collect data are themselves interventions. For the insider action researcher, data generation comes through active involvement in the day-to-day organizational processes relating to the action research project. The observations are made as a member of the organization in day-to-day interactions with colleagues and others. Not only is data generated through participation in and observation of teams at work, problems being solved, decisions being made, and so on, however, also through the interventions that are made to advance the project. (Coghlan & Brannick, 2014, 890)

From the current state needs to be known the “what”, “who” and “how”. The first step would be to gather information about the structure of the CI system. Then who are using it and how. From the desired state, the requirements given by the customer, the existing solutions (service catalog) and the restrictions are needed; both these and the current state impose.

The data for the current state is gathered through documentation walkthroughs and interviews. As always, there might some details missing from the documentation, or the implementation has evolved and the documentation was not updated, which is why the interviews are needed to improve the general view. First, the key persons responsible for administration of the CI system are interviewed. The interview method used is a semi-structured interview, where the interviewer uses drawings of the CI system as a basis for the interview. The drawings are sketched based on the documentation analysis beforehand and are “reviewed” during the interview. During the “review” casual conversation about the relocation project is upheld and more interview questions asked. The interviewer tries to keep the atmosphere positive and casual. After the interview the updated drawing is sent to the interviewee for final review. The interview situation can also be seen as an establishment of the communication channels as it could be easier to ask more information later either through email or face-to-face discussion.

Once the big picture of the CI system is gained, the users of the systems are interviewed to get the understanding about the interface between the system and the users. For this interview a semi-structured interview is used. One member from each team is chosen for this interview as each team might be using different tools based on their areas of responsibilities. Based on these interviews the use cases are sketched and sent later to the interviewees for review. Also, these interviews are used to establish channels of communication for later queries if needed. Due to the confidentiality and publicity reasons, the transcriptions of the interviews are not published.

Data for the desired state is collected from the requirements given by the customer, and from the service catalog descriptions. Email discussions or teleconference meetings are held with the service owners to gain more knowledge about the services and possible restrictions.

During this whole process the relevant literature is reviewed to widen the knowledge base of the author, and possible gaps in the knowledge of the author are filled with discussions with the experts.

All these sources of information are separately triangulated for both the current state and desired state analysis, and once the current state and the desired state are known, the GAP analysis between these states is used to identify the steps towards the desired state.

The data collected by the author is used by the team for planning and preparing the relocation. The author also observes, intervenes and collects data in the meetings and discussions with the team.

2 Virtual Enterprise

2.1 Overview of Virtual Enterprise

Kamel, Laborde, Benzekri & Barrel (2008, 1) defines the Virtual Organization (VO) as a network within the organizations (or institutions) to pool their competencies to respond to new business opportunities (or realize common educational projects). Virtual Enterprise (VE) is defined by Wikipedia (2016b) as a temporary alliance of businesses that come together to share skills or core competencies and resources in order to better respond to business opportunities, and whose cooperation is supported by computer networks. Virtual Enterprise can be seen as a particular case of Virtual Organization as Neil Kokemuller (Organization Vs. Enterprise) suggests, the Enterprise as a concept hints pursue of entrepreneurial endeavors for a profit, while organizations can be any types of social or business entities. An enterprise can be seen as a form of an organization.

2.2 Information Systems Interconnection

A system interconnection is defined as the direct connection of two or more IT systems for the purpose of sharing data and other information resources. Significant benefits to be realized through a system interconnection include: reduced operating costs, greater functionality, improved efficiency, and centralized access to data. (Grance, Hash, Peck, Smith & Korow-Diks, 2002, 2-1)

Increasing collaboration between the organizations caused by the Virtual Enterprise causes increasing interconnectivity between the Information Systems. As a consequence, each organization has to open its Information System to its partners while keeping control on its critical resources put at their disposal and preserving their security. (Kamel et al. 2008, 1)

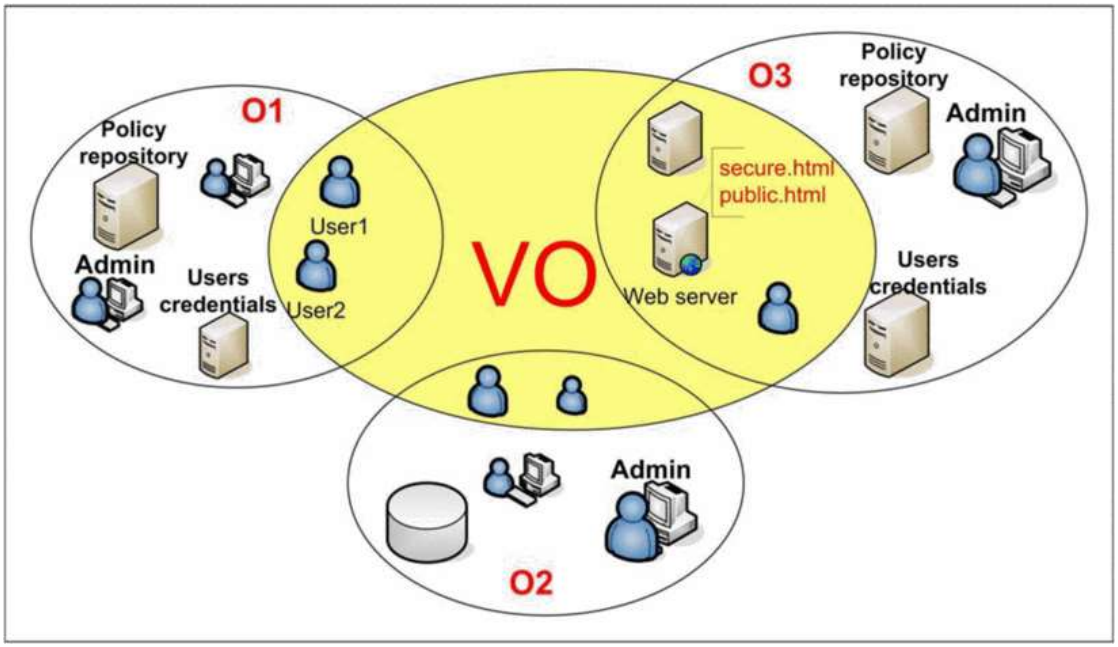

Figure 2. A Virtual Organisation network example. (Kamel et al. 2008, 2)

Figure 2 shows the example presented by Kamel et al. (2008, 2) with three organizations (O1, O2 and O3) wishing to collaborate in order to realize common projects: they decide to build a VO. Each one of these organizations has resources (human, material, etc.), capabilities and competencies that it wants to share with its partners within the VO network. In this example, O3 shares the content of a web server with users of O1 and O2. Kamel et al. (2008, 2) suggests two different approaches (centralized and decentralized) for O3 to visualize the content without risking the loss of this data nor its control of it.

Decentralized solution requires more trust from partners as each organization is responsible for authenticating their own users and providing their credentials for the organization providing the service. In Figure 2, the decentralized solution would allow 03, a resource provider, to manage only its own resources and let the remote users' organization (O1) manage its local user accounts. (Kamel et al. 2008, 2.)

On centralized solution each organization centralizes the management of users (local and remote) and resources. It does not necessitate the establishment of trust

between partners, however, it requires the administrators on the resource site have to manage remote users' accounts, in addition to managing their own resources. In a large project it could be hard and painful task, when they have to deal with a large number of users and resources. In Figure 2, O3 would consider User1, from the O1 organization, as a local user: the administrators on the O3 site would create an account for him and give him the necessary credentials (role, etc.) in order to access the resources. Thus, the user is added to the Access Control List (ACL) controlling access to the requested resource. (Kamel et al. 2008, 2.)

2.3 Securing the interconnection.

Interconnection may be worthwhile for the R&D work, but it does not come without a risk. One compromised system in the interconnected network could compromise other systems as well. Increasing interconnectivity leads also to bigger attack surface, and the impact of disruption could be devastating as working without the interconnection could not be possible. Recovering from a disaster could also be more time consuming, as there are more parties involved. Therefore, organizations are required to balance inherently opposing goals: increasing IT interconnectivity to facilitate collaboration while simultaneously mitigating exposure to IP-specific risk by increasing security (Smith, Watson, Baker & Pokorski II, 2007, 2596).

According to Grance et al. (2002, 2-1), a System Interconnection has three basic components (Figure 3): two IT systems (System A and System B) and the mechanism by which they are joined (the "pipe" through which data is made available, exchanged, or passed one-way only). The "pipe" can be established through leased line or through the Internet using less expensive virtual private network (VPN).

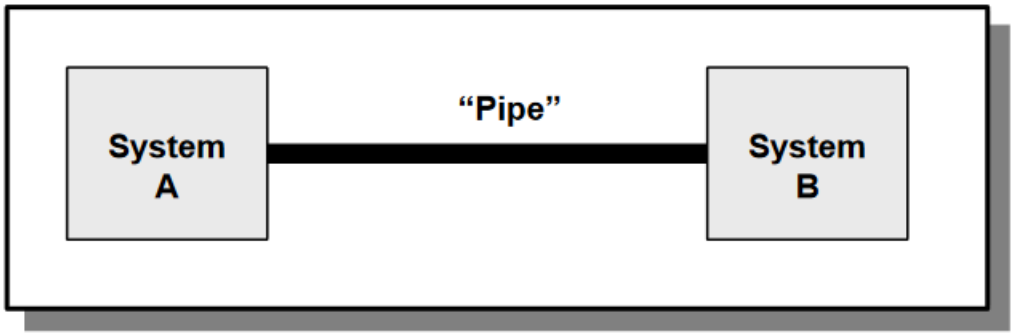

Figure 3. Interconnection Components (Grance et al., 2002, 2-1)

2.3.1 Leased Line

Leased line is a private bidirectional or symmetric telecommunications line between two or more locations provided in exchange a for monthly rent. Unlike dial-up connections, a leased line is always active. Leased lines are used to build up private networks, private telephone networks or access the internet or a partner network. (Wikipedia, 2016a.)

Leased Line can be provisioned over fibre-optic or copper circuits (depending on the bandwidth desired and the location). It provides a dedicated connection offering plenty of bandwidth, however, it typically provides no encryption or authentication as a VPN (Virtual Private Network) which provides encryption and authentication over an already existing connection – be it a Leased Line or something else. (Leased Line vs VPN – Which Technology Is Right For YOUR Business?)

2.3.2 VPN

A virtual private network (VPN) is a dedicated encrypted tunnel from one endpoint to another (Kim & Solomon, 2014, 27). In the beginning, VPNs were used to link corporate network segments over inexpensive Internet connections, using a proprietary protocol rather than expensive leased lines. Eventually, VPNs expanded

to provide connectivity from remote clients, such as laptops, into the corporate network. (Reid, 2003, 11.)

According to Roşu & Drăgoi (2011, 1), Roşu (2008) states there are three primary models for VPN architectures that can be implemented at the enterprise level:

- *Host-to-host* – used to protect communication between two computers. The model is most used when a small number of users must be online or is given a remote that requires protocols that are normally uncertain.
- *Host-to-gateway* – protects communications between one or more individual hosts belonging to a specific network of an organization. Host-to-gateway is used to allow hosts of unsecured networks an access to internal organization services such as email and web servers.
- *Gateway-to-gateway* – This model protects communications between two specific networks, such as organization's headquarters networks and organization's branch offices or two business partners' networks.

2.3.3 DMZ

According to Conklin & White (2015) the DMZ (demilitarized zone) is a military term for ground separating two opposing forces, by agreement and for the purpose of acting as a buffer between the two sides. A DMZ in a computer network is used in the same way: it acts as a buffer zone between the Internet, where no controls exist, and the inner, secure network, where an organization has security policies in place. External servers such as Web servers, proxy servers, and e-mail servers can be placed here for greater isolation and screening of IP traffic (Kim & Solomon, 2014, 24).

Figure 4 shows a layered DMZ implementation method, where the systems are placed between two firewalls with different rule sets, which allows systems on the Internet to connect to the offered services on the DMZ systems, however, prevents them from connecting to the computers on the internal segments of the organization's network. (Shimonski & Alpern, 2009, 130.)

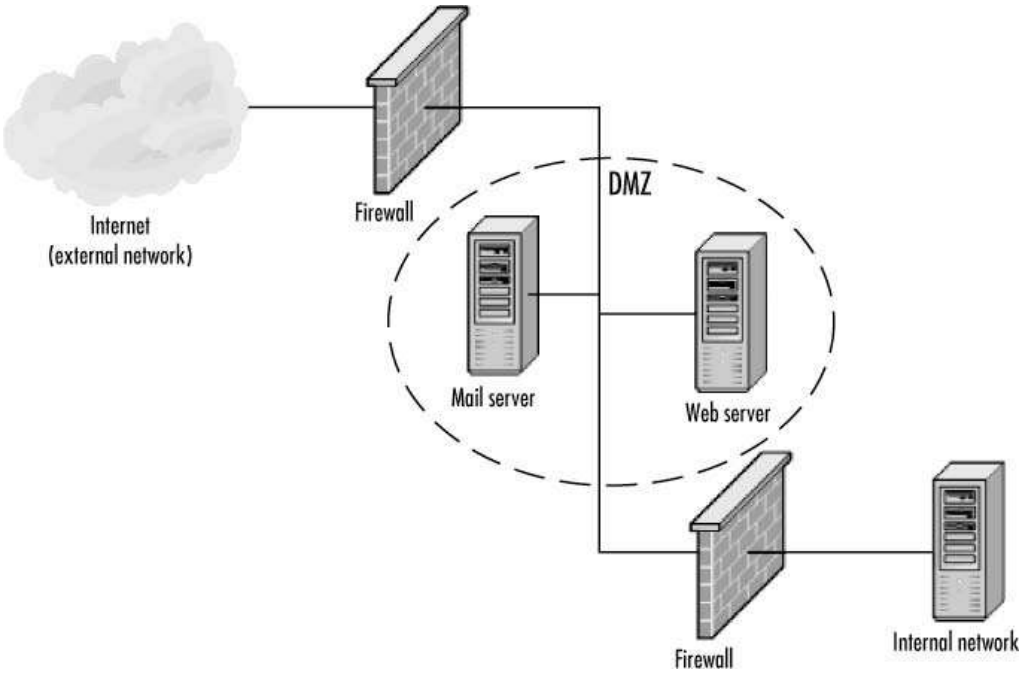

Figure 4. A Layered DMZ Implementation (Shimonski & Alpern, 2009, 130)

2.3.4 Reverse Proxy

A reverse proxy is typically installed on the server side of a network connection, often in front of a group of servers. The reverse proxy intercepts all incoming web requests and can perform a number of functions, including traffic filtering, Secure Sockets Layer (SSL) decryption, serving of common static content such as graphics, and performing load balancing. (Conklin & White, 2015, 10.)

2.3.5 Desktop Virtualization

Rouse & Madden (2011) defines Desktop Virtualization as the concept of isolating a logical operating system (OS) instance from the client that is used to access it. Desktop Virtualization conceptual models can be divided into two categories based on whether or not the operating system instance is executed locally or remotely.

Host-based forms of desktop virtualization require that users view and interact with their desktop over a network by using a remote display protocol. Because processing takes place in a data center, client devices can be thin clients, zero clients, smartphones and tablets. Included in this category are (Rouse & Madden, 2011):

- Host-based virtual machines: Each user connects to an individual virtual machine that is hosted in a data center. The user may connect to the same VM every time, allowing personalization, (known as persistent desktop) or be given a random VM from a pool (a non-persistent desktop).
- Shared hosted: Users connect to either a shared desktop or individual applications that run on a server. Shared hosted is also known as remote desktop services or terminal services.
- Host-based physical machines or blades: The operating system runs directly on physical hardware located in a data center.

Client-based types of desktop virtualization require processing to occur on local hardware; the use of thin clients, zero clients, and mobile devices is not possible. These types of desktop virtualization include (Rouse & Madden, 2011):

- OS streaming: The operating system runs on local hardware; however, it boots to a remote disk image across the network. This is useful for groups of desktops using the same disk image. OS streaming requires a constant network connection in order to function; local hardware consists of a fat-client with all of the features of a full desktop computer except for a hard drive.
- Client-based virtual machines: A Virtual machine runs on a fully-functional PC, with a hypervisor in place. Client-based virtual machines can be managed by regularly syncing the disk image with a server, however, a constant network connection is not necessary in order for them to function.

Yan (2012, 2) points out several advantages and disadvantages for a desktop virtualization. The advantages include: saving the deployment and maintenance costs

of user's desktop, centralized control promotes security and centralized management. The shortcomings can be summed up to one sentence: single-point of failure. If the server hardware breaks down, all the systems on that hardware are forced to stop. If the centralized storage has a fatal fault, the user's entire data will be lost. If the network is down or under strain, the virtual desktop is inaccessible or operates at a snail's pace.

3 The Current State

3.1 Overview

As can be seen on the high level picture below (Figure 5), the Platform (both SW and HW) and related SW components are developed by several companies. Also the customer provides several SW and HW components integrated into the platform with the components provided by the integrator company. In addition to integration, the product level testing is also responsibility of the integrator company, while the customer takes care of the system level testing. As a result, the CI machinery and other R&D systems are mainly located in the integrator company premises.

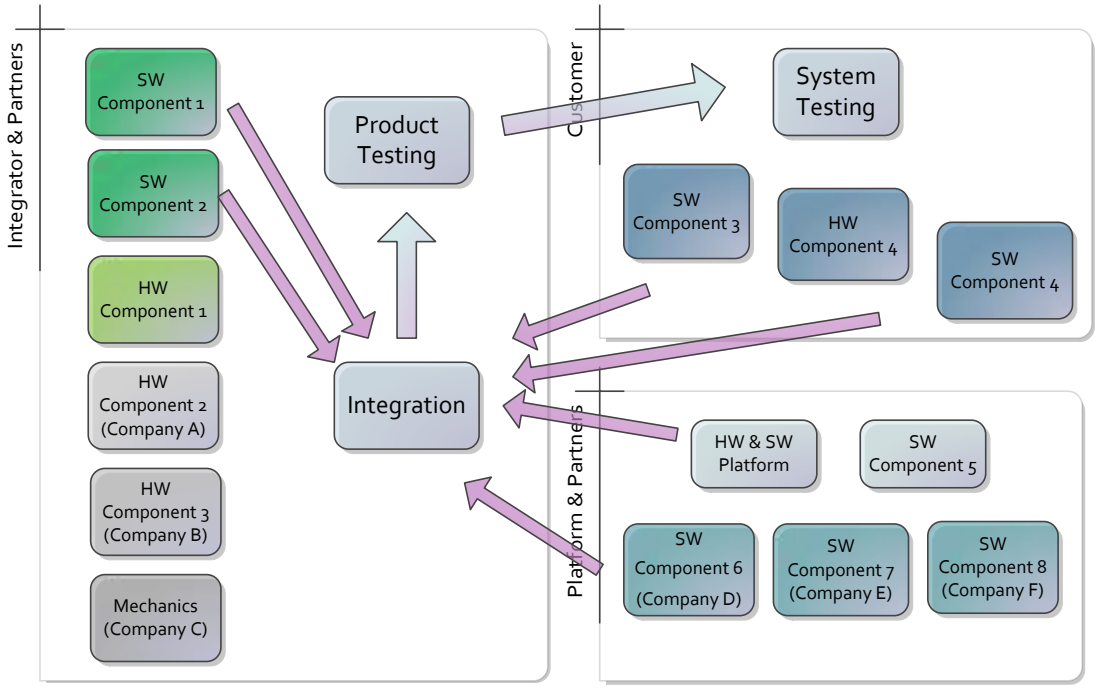

Figure 5. High level picture of current state

Figure 6 illustrates the complexity of the current CI system. Relocating it from one organization to another could prove to be quite a complex task. Factors complicating the relocation could be the incompatible IT environments, dissimilar enterprise cultures and security policies. The main challenge seems to be the difficulty of fitting the project R&D environment as such to the customer R&D environment and maintaining the level of service without any impediments to the project. Also, the fact that the relocation is only partial (HW, Middleware & teams located in other geographical locations will remain in the current premises) makes the CI system more complex and it will cause a great amount of network traffic between the companies.

In addition to the complexity, the systems were never built with flexibility or reusability in mind, therefore, the scripts used in the build systems etc. have some hard coded values and need to be gone through one by one.

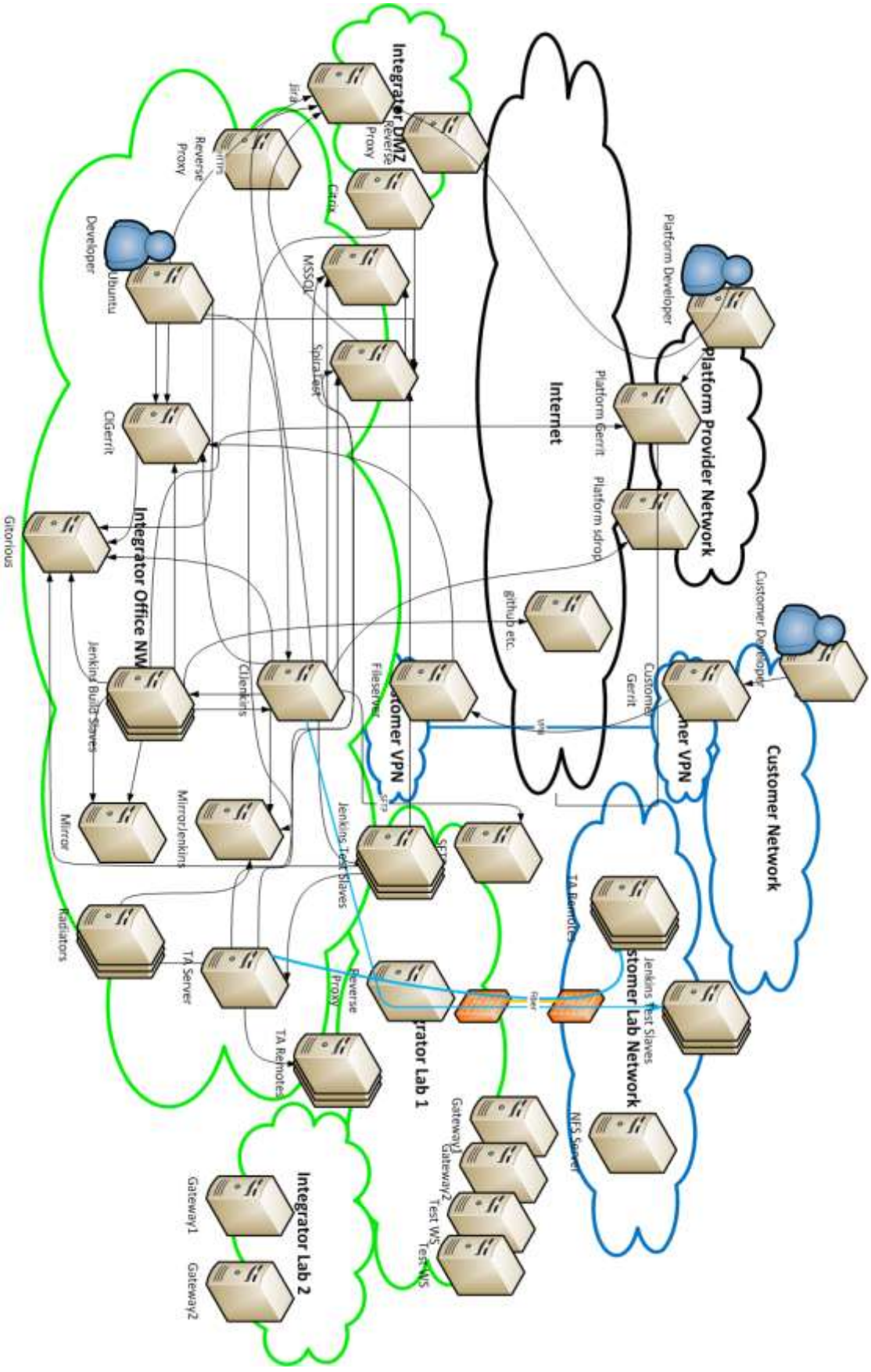

Figure 6. Initial Setup

3.2 Network Connections

As the CI system between the customer and the integrator is already interconnected, there are several network connections in place to assure the secure communication of the systems between the companies. Besides the direct connect from the Internet (and Citrix), these connections include VPN (gateway-to-gateway) and leased line (fiber). Citrix is used by customer personnel to access SpiraTest and MirrorJenkins located in the integrator network. JIRA is located in the DMZ of the integrator and is accessible through the Internet by all the partners in the program.

3.2.1 VPN (gateway-to-gateway)

The VPN connection provided by the customer is used to access the assets provided by the customer. According to customer service catalog - VPN (2015) there are several concepts to connect partners to access assets, share and work with relevant information objects. A secure IP connection through standardized connection set-ups is used for these external companies to selected and approved systems within customer company network.

These concepts include more, than just a connection. Included there are:

- Information Access Agreement
- External personnel Non-Disclosure and Access instruction
- IT Account
- Users (externals)
- Connectivity (different technical solutions available)
- Access to approved systems or applications within corporate network
- Firewall rules & management
- Incident handling by IT Service Desk

For VPN there are various access methods available to choose from. The currently used access method is a VPN (gateway-to-gateway) connection between the

companies. For this a customer controlled Firewall/VPN device was installed to the integrator premises. According to the customer security policies, this enforces a separate network in the integrator premises and the developer desktops cannot be connected to integrator and customer networks in the same time. Currently this VPN connection is used only by a limited amount of people such as the HW team to connect license servers and project managers to use systems in the customer network. For the use of this connection the user must switch the network cable physically.

3.2.2 Leased line (Fiber)

A leased line was established to connect the test laboratories of the integrator and the customer. A VPN connection was first evaluated, however, it soon became evident that the VPN could not be used because of the strict security policies applied at the customer company. It was not possible to connect the VPN network and integrator corporate network together (as already stated earlier) and secondly, the customer was very strict about the firewall rules, which made the connection impossible. To tackle this issue a fiber connection between companies was established. This fiber allows the necessary interconnection between the test laboratories, while ensuring the security. The connection is provided by the technology village infrastructure (both companies are located inside the same technology company concentration) and secured by a VPN.

The connection between the test labs is strictly secured by the firewall in the end of both connections. Through the connection it is still possible to control the test automation places in the customer test lab from the CI system in the integrator premises. Test Engineers in the customer test lab can use the connection also to connect the SpiraTest and Jira servers located in the integrator network. These connections are made through reverse proxies.

The leased line is also used for GIT synchronization between the companies. Software components provided by the customer are delivered to integrator through

the connection, and the fully integrated source code is in turn replicated to the customer servers.

These Gerrit/GIT synchronizations between the company repositories are actually against the customer security policies and established with special permit, and should not be used in the upcoming plans to interconnect the CI systems and test labs (if possible).

3.3 Tools

3.3.1 Atlassian JIRA

JIRA is a proprietary issue tracking product, developed by Atlassian. It provides bug tracking, issue tracking and project management functions. (Wikipedia, 2015e) In the project the JIRA is used for requirement, error and task tracking. Jira has inbuilt security levels (internal & external) which allows to limit the visibility of the platform developers (externals).

Jira runs on a CentOS 6 Server on a 4 core virtual machine with 16GB RAM and 200 GB HD. Jira is located at the DMZ (Figure 7) where it can be accessed by the partners and the customer test lab personnel. The personnel working in the integrator premises access the JIRA through the intranet whereas the personnel working in the customer lab use the JIRA through the leased line. The rest of the users (platform developers and other customer sites) accesses the JIRA through the Internet using a local JIRA account. Reverse Proxies are installed for each connection for information security purposes.

Not seen in the Figure 7 are the dependencies JIRA has in the corporate network including AD, DNS, NTP, SMTP and IMAP. Also, the connections from Jenkins and SpiraTest servers using SOAP/REST protocol are missing from the picture.

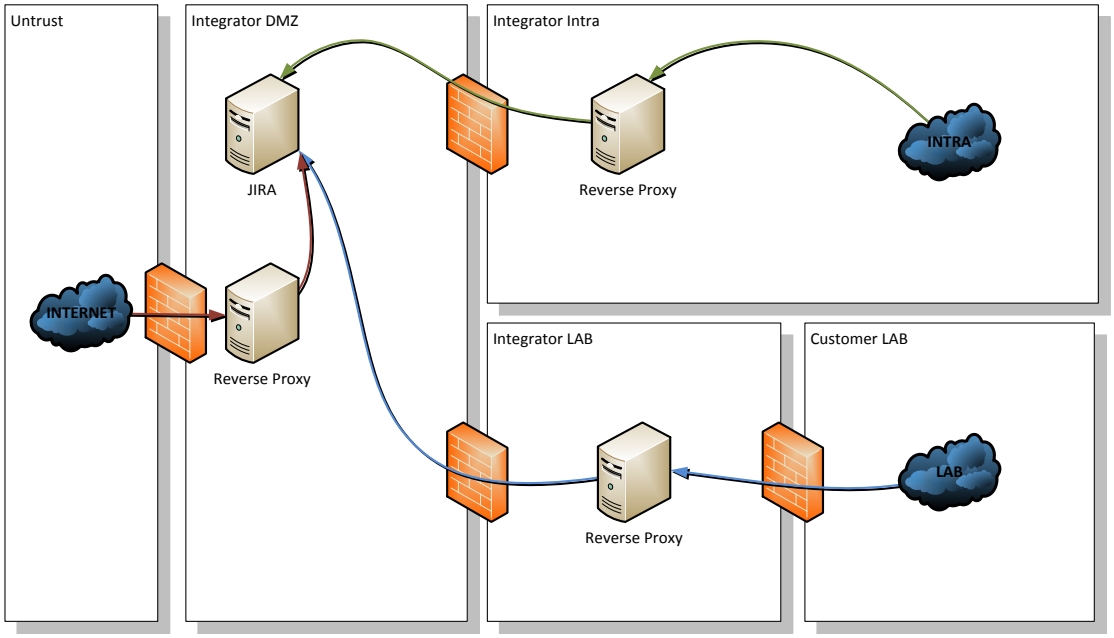

Figure 7. JIRA access diagram

JIRA users consist of local accounts and AD accounts. Local accounts are used by the project personnel not having an internal AD account at the integrator network. Local account users connect to the JIRA mostly through the internet or from the customer test lab.

Users of the JIRA are listed in Table 1. This list includes both the personnel and the system users accessing the JIRA. Integrator Users are the users accessing the JIRA inside the premises while Platform Users are the users accessing the JIRA from the Internet and the Lab Users are the users accessing the JIRA through the leased line.

Table 1. JIRA Dependencies

User	System	Protocol/Port
Integrator User	JIRA	HTTPS (reverse proxy)
Platform User	JIRA	HTTPS (reverse proxy)
Lab User	JIRA	HTTPS (reverse proxy)
CI Jenkins	JIRA	SOAP
SpiraTest	JIRA	SOAP/REST
Datamart	JIRA	HTTP/REST

3.3.2 SpiraTest

SpiraTest is a proprietary Quality Assurance solution developed by Inflectra. It can be used to manage requirements, tests, bugs and issues (SpiraTest Feature, 2015). In the project SpiraTest is used for Test Management where both manual test engineers and test automation write down the test results.

SpiraTest runs over Windows 2008 Server on a 4 core machine with 6GB RAM. MS SQL Server needed is located on dedicated database cluster, which also serves other database needs at the company. SpiraTest uses an AD for authentication and naturally a DNS and a NTP are also needed. A custom made plugin handles the test result sync between the SpiraTest and JIRA allowing the tracking of feature maturity from the JIRA.

SpiraTest is split into two databases: I&V and SW. Basically the I&V database is used by the test engineers testing at the product and system level whereas the SW database is used by the test engineers testing in the unit/module level.

SpiraTest can be accessed from the integrator corporate network and customer test lab through the leased line. Also, the Citrix can be used to access the SpiraTest through the Internet.

Table 2 represents the users and dependencies of the SpiraTest. It is used by the personnel from integrator, customer and customer test lab. In addition, systems such as CI Jenkins and Jenkins Test Slaves uses the SpiraTest for writing down the test results.

Table 2. SpiraTest users and dependencies

User	System	Protocol/Port
Integrator User	SpiraTest	HTTP
Customer User	SpiraTest	HTTP
Customer Lab User	SpiraTest	HTTP
CI Jenkins	SpiraTest	SOAP/TDS/SMUX
Jenkins Test Slaves	SpiraTest	SOAP
Datamart	SpiraTest	SQL

3.3.3 Jenkins

Jenkins is an open source continuous integration tool written in Java, which provides continuous integration services for software development. The project was forked from Hudson after a dispute with Oracle. (Wikipedia, 2015d)

In the project Jenkins is used for building and test automation. There are several Jenkins machines: the Master Jenkins (CI Jenkins), the Mirror Jenkins, the OpenGrok Jenkins, the Jenkins Build Slaves and the Jenkins Test Slaves. The Master Jenkins is used to control all the other Jenkins instances, while the Mirror Jenkins is used for the Radiators and external access through the Citrix (for customer personnel). Build slaves are responsible for the building jobs and test slaves are used to control the Test Automation Frameworks.

Build and test slaves are recognizable through the prefix in their names. Build slaves have prefix B64_ or B32_ in their name (for 64bit or 32bit processor architecture),

and test slaves are named using TST_ prefix. Build Slaves fetch the source code and libraries from the Gitorious, the CI Gerrit, the Platform Gerrit and the Mirror. Build artifacts are pushed by the Master Jenkins to the SFTP servers at the test lab and platform provider, and to the NFS share.

The Master Jenkins runs on a Linux virtual machine (Centos 6.4) as a Daemon using a local account. The virtual machine is physically located in the IM server room. Build slaves are physical PC's located in the integrator premises next to the Build & Release Team. Test Slaves are located in the test laboratories and control the test places.

In addition, there is still one more type of Jenkins in use: The OpenGrok Jenkins. The OpenGrok Jenkins is used to update the OpenGrok database. OpenGrok is a source code search and cross reference engine, which helps programmers to search, cross-reference and navigate source code trees. (Wikipedia, 2015g) Besides the database update, this Jenkins is also responsible for the test runs of Component CI test scripts and evaluation of new Jenkins plugins. OpenGrok Jenkins is located also in the IM server room on a virtual machine (CentOs 5.8), and it runs as a daemon using local account.

The Master Jenkins has currently two levels of users defined: read-only access to everyone in the project and admin rights for some persons in both the Build & Release and CI teams. User authentication is handled through LDAP.

Table 3 represents the connections between the Jenkins instances and to the services used by or using the Jenkins.

Table 3. Jenkins connections

User	System	Protocol/Port
Integrator User	CI Jenkins	HTTP/8080
Customer User	MirrorJenkins	HTTP/1503
CI Jenkins	MirrorJenkins	HTTP/1503
CI Jenkins	Jenkins Build Slaves	SSH
CI Jenkins	Jenkins Test Slaves	SSH
CI Jenkins	SFTP	SFTP
CI Jenkins	platform sdrop	SFTP
CI Jenkins	SpiraTest	SOAP/TDS/SMUX
CI Jenkins	Gitorious	SSH
CI Jenkins	CI Gerrit	SSH/29418
Jenkins Build Slaves	Gitorious	SSH
Jenkins Build Slaves	CI Gerrit	SSH/29418
Jenkins Build Slaves	Github etc.	HTTP
Jenkins Build Slaves	Platform Gerrit	SSH/29420
Jenkins Build Slaves	Mirror	HTTP & RSYNC/SSH
Jenkins Build Slaves	CI Jenkins	SSH
Jenkins Test Slaves	TA Server	HTTP/8443
Jenkins Test Slaves	Gitorious	SSH
Jenkins Test Slaves	SpiraTest	SOAP
Radiators	MirrorJenkins	HTTP/1503
OpenGrokJenkins	OpenGrok	
Datamart	Jenkins	HTTP/REST

3.3.4 GIT/Gerrit/Gitorious

Gerrit is a free, web-based team software code review tool. Software developers in a team can review each other's modifications on their source code using a Web

browser and approve or reject those changes. It integrates closely with Git, a distributed version control system. (Wikipedia, 2015c)

Gitorious is an open-source Git hosting and collaboration tool. It can be used on the official public instance, or as a self-hosted service. It offers merge requests and per-project issue tracking. It can handle both public and private repositories. (Gitorious, 2015)

In the project, Gerrit is used to review and store most of the source code of the project. Jenkins build scripts are stored on Gitorious, but replicated to the Gerrit. I&V Test scripts are mostly stored on the Gitorious with the exception of a small portion of scripts reviewed and stored on to the Gerrit. In addition, there is a Gerrit at the customer, which is used for delivery of components from customer to integrator, and for 1:1 replication of integrator repositories to the customer Gerrit. Both replications are synchronized one-way only. (GIT Sync Implementation, 2015)

Table 4 represents users and connections of both the CI Gerrit and the Gitorious. Jenkins instances fetch the source codes and test scripts from the CI Gerrit and the Gitorious.

Table 4. Gerrit and Gitorious connections

User	System	Protocol/Port
Integrator User	Gitorious	HTTPS
Integrator User	CI Gerrit	SSH/29418 & HTTP/8081
Fileserver	CI Gerrit	
CI Jenkins	CI Gerrit	SSH/29418
Jenkins Build Slaves	CI Gerrit	SSH/29418
CI Jenkins	Gitorious	SSH
Jenkins Build Slaves	Gitorious	SSH
Jenkins Test Slaves	Gitorious	SSH
CI Gerrit	Gitorious	SSH

3.3.5 Data mart

A data mart is the access layer of the data warehouse environment used to get data out to the users (Wikipedia, 2015b). In the project data is gathered from various sources, such as JIRA, SpiraTest, Jenkins and Nagios, and processed to the database. KPIs summaries are then represented from this processed data.

Data mart operates on two servers. The first server is used for Database/Web and another for the Extractor. These servers employ four CPUs, 4-6GB memory and 50-100GB HD for the Extractor and 20GB HD for the Database. As an operating system both use Ubuntu 12.04 with Linux ODBC, PostgreSQL 9.x, MSSQL Linux Driver and Python 2.7 with virtualenv.

Table 5 represents the dependencies Data mart has to JIRA, SpiraTest, Jenkins and Nagios.

Table 5. Data mart dependencies

User	System	Protocol/Port
Datamart	JIRA	HTTP/REST
Datamart	SpiraTest	SQL
Datamart	Jenkins	HTTP/REST
Datamart	Nagios	HTTP & SSH
Datamart	MongoDB	

3.3.6 Nagios

Nagios is an open source computer system monitoring, network monitoring and infrastructure monitoring software application. Nagios offers monitoring and alerting services for servers, switches, applications, and services. It alerts the users when

things go wrong and alerts them a second time when the problem has been resolved. (Wikipedia, 2015f)

The project uses the Nagios to monitor servers, such as Jenkins Build Slaves, Jenkins Test Slaves, Jenkins Master, SpiraTest etc. It utilizes four CPUs with 8GB RAM and 80GB HD.

3.3.7 Email

Project members have two different email addresses: one email address for their own company: firstname.lastname@company.com etc. and another for the program: firstname.lastname@project.company.com. Integrator provides the project email address and it is recommended to be used by all the project members. Security wise these addresses are equal – traffic is protected server to server regardless of the address. Some connections are using forced TLS and some default TLS, some have no TLS at all.

Default TLS is opportunistic – if TLS connection cannot be established the email message is sent unencrypted, as when the TLS is forced - if the TLS connection cannot be established, the message is not sent at all. No TLS means of course, no encryption. These settings only stand when sending the email from the integrator company.

3.3.8 Team site

Currently the team site is located in the customer extranet and accessed through the Internet. Most of the authentications are made using login with project email, however, some of the project members have a customer account. Login with project email currently causes some usability issues such as problems with check-out/check-in of documents or inability to search documents from the team site.

3.3.9 Developer and Test Automation Tools

Current commercial tools needed by the developers include CCStudio (Code Compose Studio) and Beyond compare. CCStudio is an integrated development environment (IDE) that supports TI's Microcontroller and Embedded Processors portfolio. Code Composer Studio comprises a suite of tools used to develop and debug embedded applications. It includes an optimizing C/C++ compiler, source code editor, project build environment, debugger, profiler, and many other features. (Code Composer Studio (CCS) Integrated Development Environment (IDE), 2015) CCStudio licenses are fetched from the license server located in the integrator network.

Beyond Compare is a data comparison utility. Aside from comparing files, the program is capable of doing side-by-side comparison of directories, FTP and SFTP directories, Dropbox directories, Amazon S3 directories, and archives. It is available for Windows, Mac OS, and Linux operating systems. (Wikipedia, 2015a) Beyond Compare uses a local license, so it has no external dependencies.

In addition to the tools that the developer uses for the software development, there are tools used for test automation including TTP and OpenTTCN. TTP (Test Tool Platform) is a proprietary tool for scheduling the test automation runs to test places and marking the results to the SpiraTest database. It has two components: TTPRemote is located in the test places and controls the test place, and communicates with the TTPServer, which is responsible for scheduling and writing down the test results to the SpiraTest. TTPRemote is currently located in the integrator network, so there currently is an external dependency from the test lab to the integrator network.

OpenTTCN Tester 2012 is a sophisticated TTCN-3 editing, test execution, debugging, and visualization, analysis, and test automation tool. OpenTTCN Tester 2012 can be used to build in-house test systems based on standardized TTCN-3 and TTCN-2 testing languages and ASN.1, XML Schema, and CORBA IDL data description languages. (OpenTTCN Tester 2012 Tour, 2014) OpenTTCN is a proprietary tool and

needs to access the license server. There are some local licenses for the OpenTTCN though, however, those instances of OpenTTCN cannot be updated.

3.3.10 Backups

According to Program Security Description (2015) the backups are taken regularly. Full backups are performed during each weekend and the incremental backups are performed after workdays. The first and third full backup of each month is stored permanently, while other full and incremental backups are stored for 6 weeks, after that the backup media is reused. The tapes are stored in a specific room with a proper physical security controls (such as access control, camera surveillance etc.) implemented.

3.4 Use Cases

Key persons from the integrator company were interviewed in order to gain an understanding how the systems interact with each other, and how the personnel is using the systems. Use cases were modelled based on these interviews.

3.4.1 CI Machinery

Figure 8 represents the high level view of the CI Machinery used in the project. First comes the SW Development where the actual source code is being written and pushed to the Gerrit for a review. Once in the Gerrit, the task is reviewed by a fellow developer and if approved, the task is submitted to the repository of the particular component, where the Component CI builds the task and executes the Component CI Test Automation. Once the Component CI is finished and the task approved, it is promoted to the mainline candidate where the new build is executed by the Product CI, including the other new tasks promoted by the Component CI. Once the Product

CI is successfully finished, the mainline candidate is promoted to the verified mainline for daily build and testing.

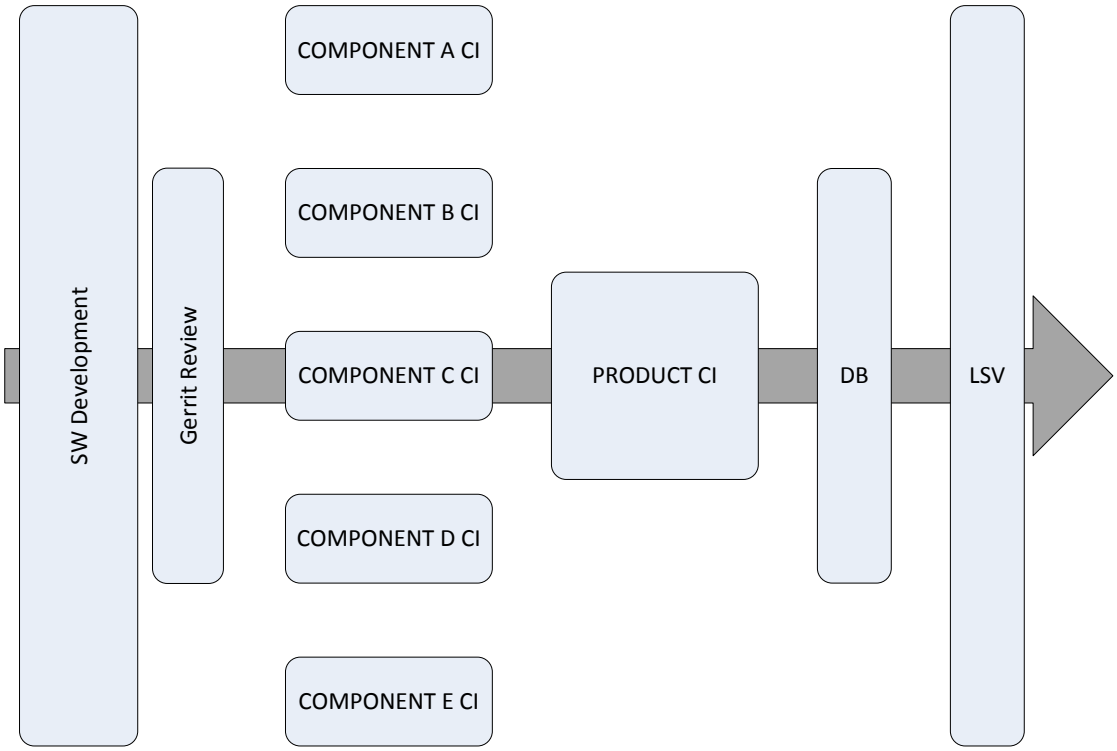

Figure 8. High level picture of CI Machinery

3.4.2 Middleware Component CI

Middleware can be seen as an abstraction layer between the hardware and the upper level software components. It hides the details of hardware from upper levels and production testing modules, and includes kernel drivers and middleware libraries. It operates in both kernel space and user space. The Middleware Team is responsible for developing this component.

When studying the usage of the R&D systems, two separate roles from the Middleware team were chosen. These roles are the Developers and the Test

Engineers. Developers develop the Middleware component and Test engineers test and/or create automated test cases. Test engineers are also responsible for the maintenance of the test places while Build & Release, and CI teams monitor the CI and the source code repositories. Figure 9 represents the interaction between the team members and the Component CI and other R&D systems.

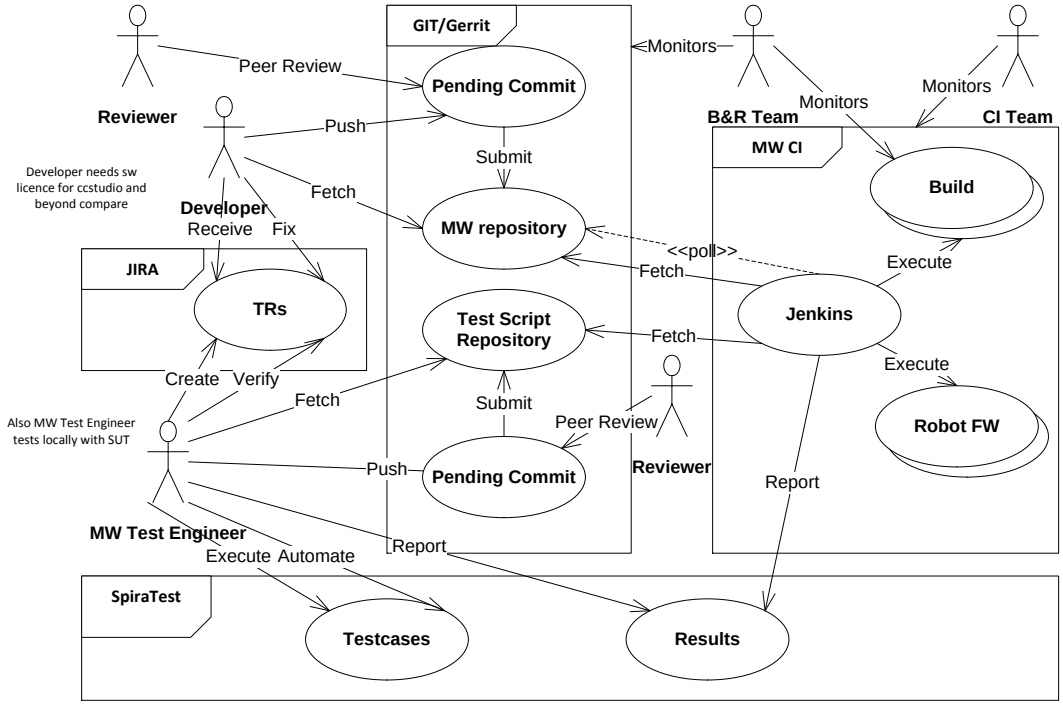

Figure 9. Middleware Component CI Use Case Diagram

MW Developer (Error Fix)

This use case describes the typical case where a Test Engineer has created an error to the JIRA. Once the error has been assigned to a developer, he/she fixes the error and pushes the commit to the Gerrit. Commit is then reviewed, built and automated tests are executed.

The following steps can be identified:

1. Test Engineer creates an error (JIRA) which is assigned to the Developer
2. Developer Fixes the error and pushes commit to the Gerrit
    a. Fetch the latest mainline
    b. Debug the error on the HW if needed
    c. Fix the error and build a test image
    d. Test the fix on the HW locally
        i. OR developer test job at Jenkins
        ii. OR sandbox build to lab test engineers
    e. Push the commit to the Gerrit
3. Reviewer does the peer review for the commit
4. The commit is submitted to the MW Repository
5. Jenkins initiates a build for the changes
6. Jenkins fetches the test automation scripts and initiates test execution
7. Results are reported to the SpiraTest (or Jenkins).
8. Test Engineer is alerted through email about failed test run.

MW Test Engineer

Testing is mostly automatized in the Middleware team, so the main task for Test Engineers is to automate the test cases. In addition to test automatization, the test engineers are responsible for the maintenance of the test places. Should troubles arise in the test execution, the test engineers would try to sort out (with the help of the developers), whether the issue is in the test environment, or is the middleware software component to blame.

Steps to test case automation are:

1. Test case to automate is assigned to the TA Engineer (JIRA)
2. A Test case is written and commit is pushed to the Gerrit
    a. Test content is read from SpiraTest

        i. OR Developer defines the Test Case & it's later added to SpiraTest
    b. Latest scripts are fetched from the Test Script Repository
    c. A Test Script is written
    d. The Test Script is tested in the HW locally
        i. OR the test script is tested on the CI system
    e. The Test Script is pushed to the Gerrit
3. Reviewer does the peer review for the commit (Test Script)
4. Commit is submitted to the Test Script Repository

3.4.3 SW Component CI

SW Component is a higher level component in the software stack. This component is developed in the branch office. The Component CI system is very similar to Middleware CCI as Figure 10 indicates. There are more Test automation frameworks in use, which have external dependencies (not visible on the picture). These dependencies include the TTPRemote, the client part of the TTP, whereas the server part (TTPServer) is located in the corporate network. TTCN-3 in turn is based on the OpenTTCN tool and has a license server located also in the corporate network.

Like in the case of Middleware Team - also from this team two roles were chosen to study the usage of the R&D systems. Similarly these roles are the Developers and the Test Engineers. Developers develop the SW component, and the Test Engineers test either manually or develop test scripts for the test automation.

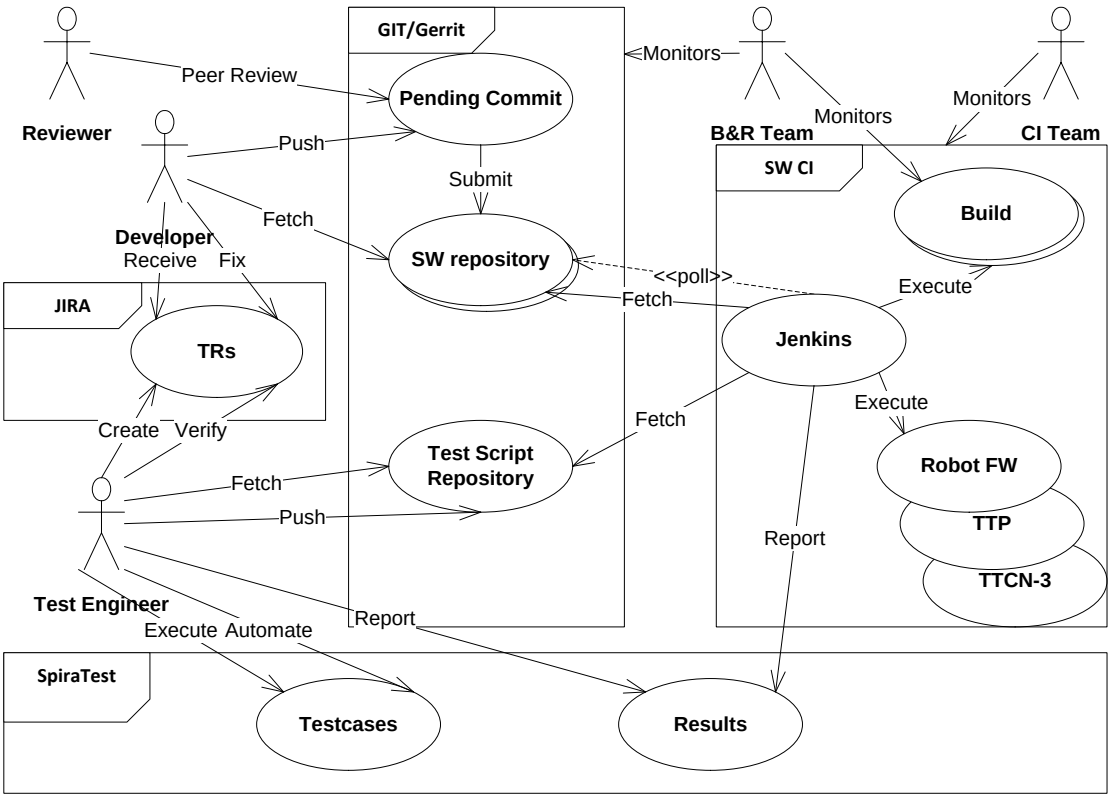

Figure 10. SW Component CI Use case diagram

SW Developer (Error Fix)

This use case describes the typical case when the Test Engineer has created an error to the JIRA and the Developer fixes the error and pushes the commit to the Gerrit. Once the commit is reviewed, it is built and test automation is executed.

The following steps can be identified:

1. Test Engineer creates an error (JIRA) which is assigned to the Developer
2. Developer Fixes the error and pushes a commit to the Gerrit
    a. Fetch latest mainline
    b. Debug error on HW or Simulator

    c. Fix the error and build a test image
    d. Test on a HW or Simulator
        i. OR developer test job at Jenkins
        ii. OR sandbox build to lab test engineers
    e. Push the commit to Gerrit
3. Reviewer does the peer review for the commit
4. The commit is submitted to the EOAM Repository
5. Jenkins initiates build for the changes
6. Jenkins fetches the test automation scripts and initiates test execution
7. Results are reported to the SpiraTest (or Jenkins).

SW Test Engineer

Testing is for most parts automatized in the SW team as well, so the main task for the Test Engineers is to automate the test cases. Test Engineers are also responsible for the maintenance of the test places and if there are any problems in the test execution, the test engineers try to solve these problems.

Steps to test case automatization are:

1. A Test case to automate is assigned to the TA Engineer (JIRA)
2. A Test case is written and a commit is pushed to the GIT
    a. Test content is read from the SpiraTest
    b. Latest scripts are fetched from the Test Script Repository
    c. The Test Script is written
    d. The Test script is tested on the CI system
3. The commit is submitted to the Test Script Repository

3.4.4 Hardware

The main responsibility of the Hardware (HW) team is to design the HW of the product. The team consists of personnel from several companies. HW Team is not as tightly integrated into the CI System as Middleware and SW teams are, however, they also use some of the R&D systems in the project. GIT is used, for example to save test scripts written by HW verification engineers. JIRA is used for errors, tasks etc., and SpiraTest is used for test cases and test results – in the same way as the other teams use them.

The test scripts used by the HW verification engineers are written in TTCN-3, which means similar license server dependency with the OpenTTCN as the SW Test Engineers have.

When studying the usage of R&D systems, only one role from the HW team was chosen. This role is the HW verification engineer. HW verification engineers are responsible for testing the HW from various perspectives. Even though the HW engineers were not chosen for the use cases, it still needs to be remembered, that they use tools from customer through the VPN connection, and they save some of their outputs to the GIT. HW Engineers also use MW test place for RF calibration; therefore, they should not be forgotten when planning the move of the CI infrastructure.

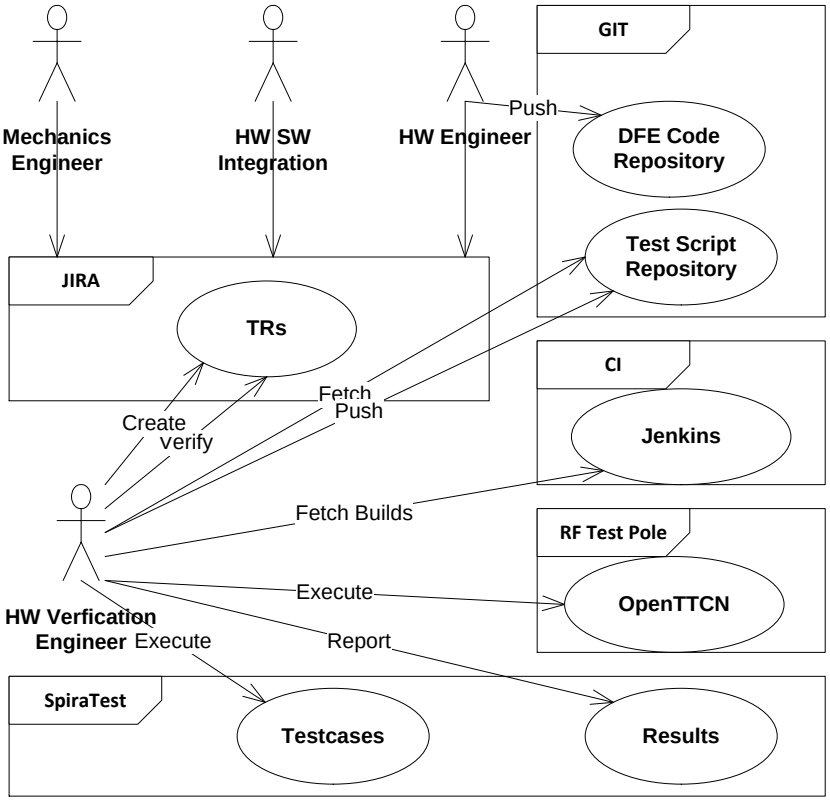

Figure 11. HW Use case diagram

HW Test Engineer

This use case describes the way HW verification engineers use the R&D systems, when automatizing the test cases. The content of the test cases is based either on the external (3GPP) or internal requirements.

Steps to test case automatization are:

1. A Test case to automate is assigned to the TA Engineer (JIRA)
2. Test content is read from 3GPP specification
    a. OR test specification on the team site
3. Latest scripts are fetched from the Test Script Repository
4. A Test Script is written
5. The Test Script is tested in a HW
    a. OR the test script is tested on the CI system
6. The Test Script is pushed to the GIT

4 The Desired State

4.1 Overview

Figure 12 represents the high level view of the desired state given by the customer. Compared to the current state (Figure 5), the integration is moved from the "integrator" to the customer. In practice, the CI machinery is relocated from one organization to another, while the persons working in the project remain the same and most of them need to move to the customer premises. The move of the personnel is not covered in this study, however, the current plan is to leave the HW team and the Middleware team to their current premises, while the Integration & Verification, Build & Release, CI and management teams moves to the customer premises. Also, teams located in other geographical locations will not be affected.

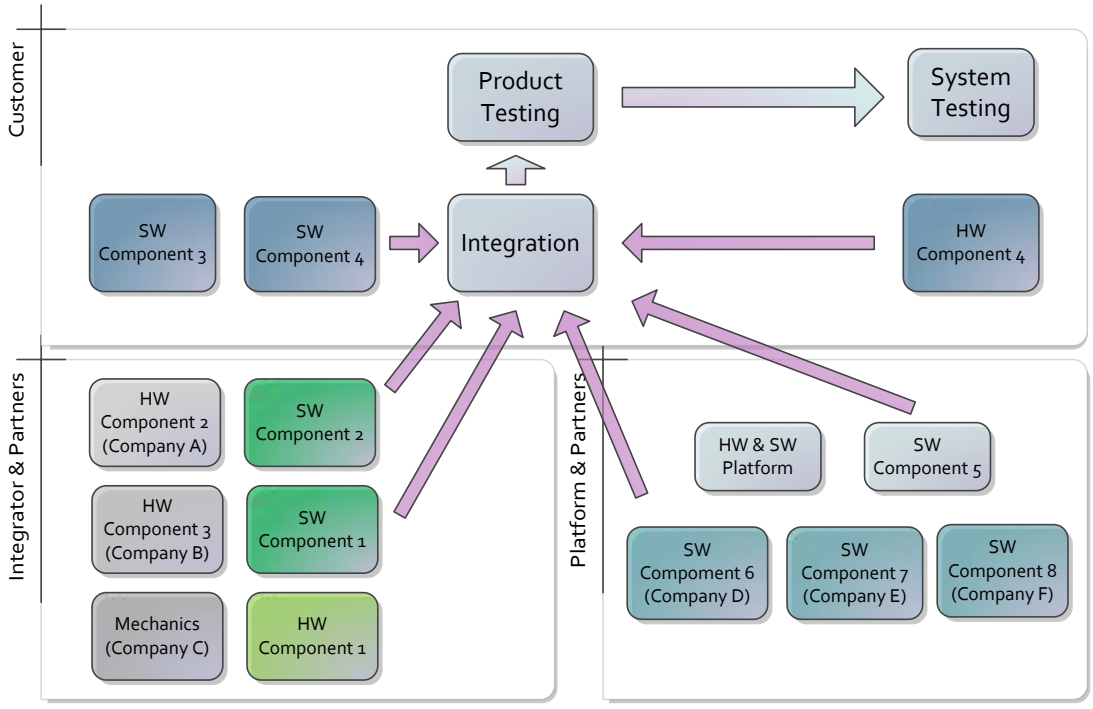

Figure 12. High level picture of target system.

Because of this CI Infrastructure relocation, there is a need for solutions, which allows the development and verification work to continue from integrator premises.

Figure 13 describes the current best understanding of the forthcoming CI infrastructure in the customer premises. As can be seen in the figure, the developers will push their commits directly to the repositories in the customer Gerrit. Commits would be built by the customer CI system and the build artefacts would be delivered to integrator using the ARM and QNAP NFS shares. Test Automation Frameworks in the integrator premises would be controlled by the customer CI system.

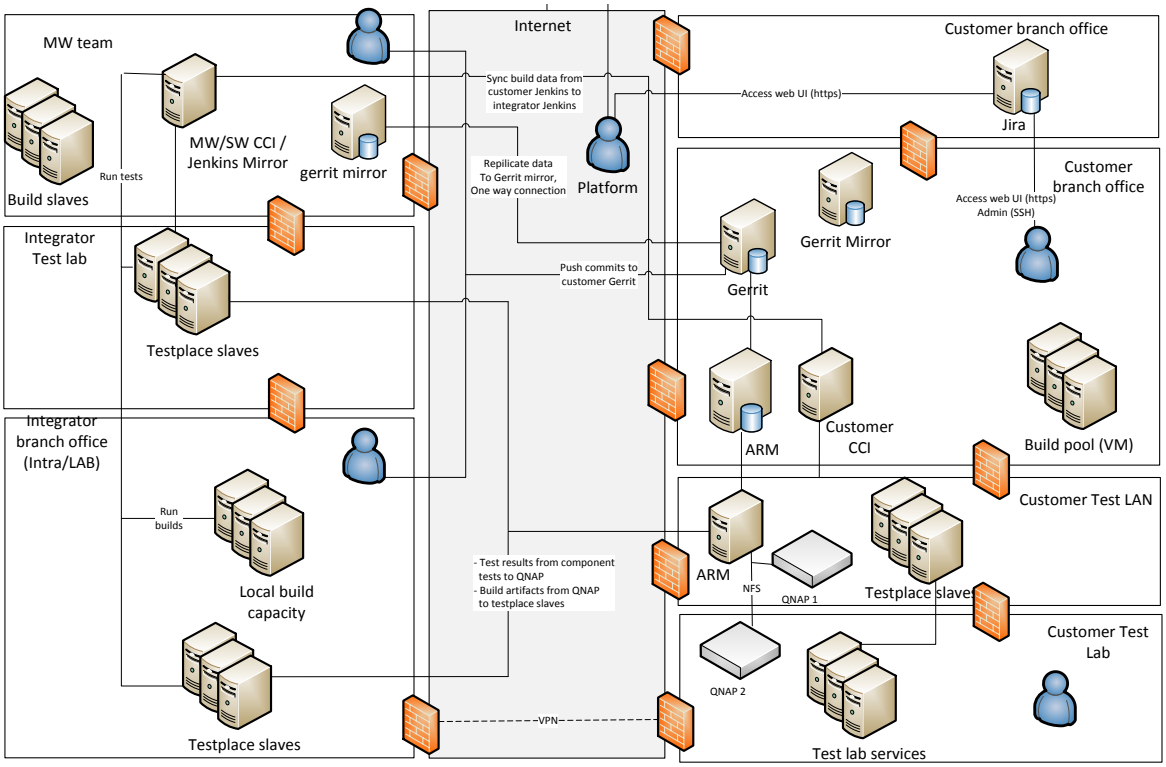

Figure 13. Initial plan for CI Infrastructure in the customer premises

4.2 Network connections

The customer service catalog has two different (feasible) solutions that enable outside users to work in the customer network. These solutions are the Citrix based

Virtual Desktop and a VPN (Host-to-gateway). Two requirements guide the choice between these two. The first requirement is to use external accounts as much as possible, and another is to apply the partners to use their own laptops instead of using the workstations given by the customer. Both these requirements aspire cost reductions as no additional workstations need to be acquired, nor a complicated processes for internal user accounts need to be carried out.

4.2.1 Virtual Desktop (Citrix)

Customer has a Citrix based remote desktop service that provides client independent access to applications and business systems without installing the applications locally on the client. It can be divided into two different products - Non-persistent and persistent virtual disk. The fundamental difference between these is the ability to install applications – with the non-persistent version users are not able to install applications, while using the persistent version users are allowed to manage their virtual desktop and install applications. (Virtual Desktop User Guide, 2014)

In order to use Citrix user needs:

- Citrix Receiver installed
- Existing home drive
- Strong authentication through Soft Token (MobilePass) or Hard Token.

Citrix based virtual machine has some limitations though:

- It is not possible to use local storage (HD, USB stick etc.) from untrusted devices. Untrusted devices are basically all non-customer controlled devices. This is by design/security feature.
- Files saved outside of home drive are erased during the restart of the virtual machine (this applies only to non-persistent version).
- Internal user account is needed to use Citrix (There was also ongoing activity to use an external user account, but it was cancelled).

Two different usage scenarios can be identified for Citrix:

- Citrix would be used for all the activities including: development, testing, test automatization and use of the R&D systems (JIRA etc.).
- Citrix would be used only to access R&D systems like: JIRA, SpiraTest, Jenkins etc. Development (both the SW and test scripts) would be handled on a local PCs and the output would be sent through other connections (VPN or leased line).

The plan was to use the Citrix from both the integrator and the customer premises using the personnel's own laptops. When in customer premises, the user would first take a VPN connection to the integrator network. This would allow the security updates to be installed and emails to be read for both the integrator and the project emails, which might have caused some problems such as:

- VPN server capacity limitations at the integrator
- Everybody not necessarily have a laptop
- MobilePass could not be used as it needs internal account, so the Hard token would be needed instead.
- Capacity of the WLAN at the customer premises might not be enough, so a wired internet connection might have been needed.

And if the user had been in the integrator premises (Middleware & SW), there would have been problems with:

- License server connection for middleware developers (CCStudio), unless the licenses could be fetched from the customer license servers.
- Ability to debug in HW

When evaluating the persistent version of Citrix, there were occasionally some hick-ups, such as problems with printers, or the connection was lost, or there were some problems to connect in the first place. Some of these problems were caused because of the insufficient resources in the WLAN access point, however, this was later fixed

by adding a new access point. Also, when using several Virtual Desktops or Remote desktops within each other caused a notable delay, which might have been frustrating for the users. This delay could have been decreased by using wired internet connection instead of the WLAN and by not circulating the connection through the integrator VPN.

During the evaluation of the version of Citrix which allowed the use of external account, the development program was cancelled due to the security reasons. This meant the only way to access Citrix would be using the internal account. As it was seen as too costly for the program, a host-to-gateway VPN was represented as an alternative. The Citrix was ditched as the main way of connecting to the customer network. However in some circumstances the Citrix can still be considered and that is why this option is still enclosed in this report.

4.2.2 VPN (Host-to-gateway)

Host-to-gateway (or personal) VPN is a part of customer service catalog provided for partner access. It involves a VPN client installed to each developer PC enabling a direct network connection to the customer network (through several firewalls) and as such, the client used to connect must meet minimum set of security requirements to be considered trusted:

- Up-to-date version of AV/antimalware SW and Hard drive encryption.
- Customer user account
- Strong authentication through hard- or soft token.

In order to user personal VPN, the user needs to have at least an external user account to customer networks as well as a hard token for strong authentication. Each user needs to have the VPN enabled and access to needed services configured by the customer.

According to guidance (Service Catalog – VPN, 2015) provided for the Personal VPN, the current client is Windows based only, including Windows 7 and Windows 8. Current platforms that are not supported are listed as follows: Apple IOS, Android, Mac OS X, and Linux.

This lack of support for Linux OS might cause problems for the developers as the development currently takes place in the Linux workstations. Another obstacle for Linux desktops is the requirement for hard drive encryption, which is currently not present in the integrator development machines – only laptops.

The use of personal VPN from integrator laptops using Windows might be feasible; however, currently the Hard Drive Encryption solution used by the integrator is not yet approved by the customer IT department and is currently in consideration.

Other open question the personal VPN is the usage of email and communicators. When connected to VPN, the Outlook email client or the Office Communicator cannot be used. Emails could be read through the web client while connected, or the VPN connection could be disconnected (and connection established to the integrator VPN) from time to time in order to read the emails. It was also mentioned in the VPN Client Access User Guide Hardtoken (2011) that there are some compatibility issues with the Cisco VPN Client, which happens to be the client Integrator is using for VPN connections.

4.2.3 Leased line (fiber)

The fiber connection would still be needed if a part of the test laboratories are left to the customer premises (Component CI). The firewalls in both ends of fiber would just need to be reconfigured to allow the needed traffic (basically the current setup reversed). This option is not fully investigated and should be considered depending on the choice made about the location of the Component CI.

4.3 Tools

4.3.1 Atlassian JIRA

Location for the JIRA is still somewhat unclear; however, there is a process ongoing with the customer IT department where the most viable solution should be determined.

According to current understanding, there would be need for three different routes to access the JIRA: one for Personal VPN users, which would include a VPN tunnel and reverse proxy deployed for security. They would use an external account for authentication. Platform developers would access the JIRA directly from the Internet through reverse proxy, but exact details of this approach are still open. Customer internal users would access the JIRA directly from the customer network.

It is utmost important to get JIRA access for all the partners in the consortium. While there still might be some obstacles in the method described above, one issue that is not yet considered is for also Platform developers and partners to use the Personal VPN with external account and more strict rules in the firewall (that would prevent every other connection). The use of an internal account for this might also prevent the case of unused JIRA accounts as there would be an existing process for this.

While there is a process ongoing in the customer IT department (it is expected to take some months), an alternative solution for JIRA is to leave it to the integrator network for a while and move everything else to customer premises. If this approach is taken, the effect to the connections between the JIRA and the SpiraTest or the Jenkins needs to be considered. It is expected that it should be possible to connect these systems; however it is not tested yet. The fact that the JIRA is in production use might cause some risks to testing this system.

It is still not fully clear if the customer IT department can provide the solution needed. There are still some open questions related to the process:

- Will the customer IT department provide a turn-key solution, or is there planning/implementation?

    - DMZ Zone setup (Firewall policies)
        - Access between The JIRA and the SpiraTest (and Jenkins)
        - Access from Internet, Intranet and Personal VPN
        - Authentication from AD?
    - Order process for the servers
- Who is responsible for the administration of the JIRA Server (both the server and the content? The JIRA is quite a haystack and needs constant administration. It should be considered if the personnel currently looking after JIRA can still be used for administrating it even after the relocation; otherwise there is a high risk for deteriorating content.
- How is the database replication from the integrator to customer handled?
- How is the username switch from integrator AD to customer AD handled? Also the old local accounts should be cleaned up.
- If the JIRA were located in the integrator network and SpiraTest in the customer network, then an open question would be if it were be possible to secure the SOAP/REST connection with HTTPS (Can the Reverse Proxy handle it?)?

4.3.2 SpiraTest

For SpiraTest there are two options: the entire SpiraTest would move to the customer network or the SpiraTest would be split in half. The half with lower level (component) test cases would stay in the integrator premises as it is, and the higher level test cases would go to the customer premises. The split would be done by duplicating the SpiraTest database from integrator to customer. The test results would be marked respectively to relevant databases and the results of irrelevant test cases would not be updated anymore. These irrelevant test cases would be either cleaned up or just ignored.

What would be the most practical option would depend on the location of Component CI and the accessibility for users. Accessibility does not seem to be an issue here, as the Personal VPN should enable to use the SpiraTest also from the

integrator premises, however, the location of the Component CI (Middleware & SW) should be taken into account.

Anyway, the SpiraTest at customer would most likely be located in the corporate network installed on Virtual Machine (4 core processor, 6GB – 8GB RAM and Windows 2008/2012 Server). SQL database could be located in the MS SQL server cluster which should be available.

4.3.3 Gerrit/GIT

The same way as the SpiraTest, the Gerrit has two options depending on the location of the Component CI. If the Component CI is located in the customer premises, also the Gerrit should be located there; if the Component CI is located in the integrator premises, then the Gerrit should be split and one Gerrit planted to the integrator and another to the customer Gerrit central. The synchronization could be “one way sync”, as there should not be anyone at the customer contributing.

The first option would lead to the question:

- How is the development handled as the Personal VPN might not be compatible with the Linux desktops (as stated in the chapter 4.2.2 - VPN (Host-to-gateway)).

The second option would also leave open questions:

- How could the mirroring between the local Gerrit and customer Gerrit be handled?
- How would the Component CI building and test automation handled (locally or from customer premises)?

These open questions are relevant, because in the past there have been difficulties to get the ports open from firewalls in order to allow the interaction between the R&D Systems at the integrator and customer networks.

A few words about the Gerrit at customer premises: There is a Master server for pushes and Slave server (mirror) for clone/fetch (read-only). VPN usage should also be possible; however, this should be confirmed.

4.3.4 Email

The external account that the customer is providing, does not include email address. So there is a need for the project's own email address – otherwise everybody would need to use the address of their own company. As the project members (particularly subcontractors) are coming and going, the author would recommend the use of project address also in the future. This way it is easier to handle the mailing lists and mail chains so that there will not be any email sent to someone who has left the project already. It could be considered though, if the external accounts were be used for similar control for limiting the email exposure to the parties already left the project.

When connected to the Personal VPN, the email should either be read through the webmail or the Personal VPN connection should be disconnected for email reading (when using the integrator laptops).

4.3.5 Teamsite

All the project user accounts used should be updated to external accounts, which should improve the usability and security of the team site. Once there are no more project user accounts it could also be considered if the team site could be moved out of the extranet and if the access could in the future take place through the Personal VPN and corporate network only.

4.3.6 Backups

The systems are currently backed up on regular basis and this should be the case also after the relocation. Depending on the server location, the backups are taken care of either by the service providers, or if the servers are located in the local customer premises, the backups need to be taken care of by project personnel.

What happens to current backups should also be decided. Currently the tapes used for backup the project data might include data from other projects, which makes deleting this data difficult. It would be recommended not to delete the data immediately, as there might be need for those backups later.

5 Summary

The move is a very complex project and making it as simple as possible would be a good bet, which means that we would forget the services the customer is providing such as extranet and similar if possible. The easiest way would be to build as similar an environment as possible to the customer local premises as a first step. Once everything is working, and the project work can continue we would start to think moving the systems one-by-one to the corporate cloud etc. This would decrease the risk for the project remarkably. In this case – if the JIRA were to be located in the customer local branch office, it could be possible to get the access there also for the platform development personnel through the Personal VPN. Expenses of this option might be more than expected, however, it could be considered.

For the same reason, also moving the Middleware team to the customer premises could be a good idea, which would also decrease the complexity of the CI system and reduce the connections between integrator and customer premises. This would not help the similar situation with the SW team though, as moving the team from other site is not an option. Another option would be to increase the usage of the leased line and improve its performance. The possibility that the Middleware team is unable to use the Personal VPN (no Linux support) from their development PCs (no encryption) could also favor the usage of the leased line (this, of course if the team is not moving).

Information security risks should be managed if not yet done so, which would include the information security risks and the risks for physical security of the IT equipment in the customer premises. According to Information Security Risk Management document (Information Security Risk Management, 2015), the Information Security Risk Management must be considered if there is a risk of significant Business Impact. Example are as follows:

- When there is a change or an activity where information security risks can emerge that have an impact on ability to carry out the strategic plan

- When new information, or a new information processing system, is introduced, either in terms of a project, in- or outsourcing, merger & acquisition, or through other circumstances
- When internal or external activities will lead to a change in the existing information processing environment
- When the loss, disclosure or denial of access to an asset would cause danger to any person, damage to Company, partners or customers, a loss of business or impact on revenues, moral or Company's reputations

The IT assets the customer has would be managed as part of this Information Security Risk Management Process.

Possible deviations from security policies need to be explained well and perhaps some sort of plan given, how those could be remediated in the future. The customer has an exemption process for IT Security policy deviations, which might help if needed. More details can be found in security exemption guidelines (IT Security Exemptions, 2015).

If some of the R&D systems are going to be placed in the Customer local branch office premises, there might be a need to define the information owners, listing the assets and assess the risks. Also, the ISMS might need to be defined. All the employees working in the customer premises should be educated in the ways of the customer. These trainings should include matters such as:

- Security Training (usage of ID card and whom to contact if ID card is lost, access control systems etc.)
- Basic Safety instructions
- Introduction to customer tools (IT Support etc.)

The JIRA currently has plenty of unused local accounts. These users are piled up from the beginning of the projects and many - if not most of them have left the project already. Why the check-out process has been ineffective here is because the check-

out process only affects the personnel working at the integrator premises, and there have been great amount of temporary personnel, for example from other customer sites working for the project. This can be seen as a major security risk for the project, as some of the local accounts seem to be external and there is no confidence that these employees are still working for the project. The excess local accounts should be cleaned up or disabled, perhaps before the move, so there would be less accounts to convert to customer accounts. A good starter would be to disable all accounts that have not been used after last summer. Once the relocation is done, the customer should define a more solid check out process to prevent unused local accounts.

6 Conclusion

I was hired to customer organization as a consultant to figure out possible security implications the CI system relocation project could cause. Later this assignment transformed more to participation in the planning of the relocation project – not just from security perspective but also solving out if can be done in the first place as required. The primary goal for the study was to enable the relocation of the CI system from one partner organization to the customer organization, and the secondary goal was to meet the requirements given by the customer for technologies to be used in the desired state (Private Cloud and Virtual Desktop). Given time frame was four months (the length of the consulting contract).

The study separated to five different phases, where I needed to solve:

- What is the current CI system like?
- Who are using the CI system?
- How the CI system is used (use cases)?
- What is the desired state?
- How to reach the desired state?

In the first phase I solved out the structure of the existing CI system. For this I studied the existing documentation and interviewed people maintaining the system. These interviews were my first interviews and allowed me to practice for the forthcoming interviews with the users. The interviews went quite well apart from one mistake I made in my first interview. I forgot to ask permission for recording before the interview. Fortunately, the interviewee did not seem take offense, however, but it was quite embarrassing for me.

Before interviews I had done some sketching about the environment, which we went through with the interviewee. In addition, I had prepared some additional questions for the occasion. The sketching of diagrams when interviewing proved to be a good idea, as it made the situation more relaxed and the discussion more casual. And in the end, I think this was the best way to get an overview of the system.

Next I needed to find out who are using the system in order to interview how they are using it. For this I contacted the each team leader to get best candidates for the interview. I prepared some questions beforehand and during the interview I showed the diagram about the current setup. This time I did not forget to ask permission to record the interview. After the interviews I sketched the use case diagrams and sent them through email for interviewees to review.

So far everything was straightforward and went well. The difficulties came later when planning the desired state. As stated earlier, the customer had given two requirements. The first one was to use a private cloud that was still under construction, however, it should have been ready when the relocation was planned to happen. The second requirement was to use a virtual desktop based on Citrix as the main connection method to the customer network environment. Both of these requirements were later found unfit for the relocation project as the private cloud project was late and was not expected to be ready when the relocation was planned to happen. The private cloud would have caused many other issues as well, so maybe this was better for the project. The virtual desktop was discarded, because it could not be used without internal account, which would have been too expensive for the project.

Planning the desired state with the above requirements took quite a substantial amount of time and was a quite tiresome process. As seen in the Figure 6, the network of systems is quite complex, where everything affected everything. In the planning process, where we found a seemingly good place for some server, we soon found out, that it could not be installed there because of some other dependency could not be met. This went on some time and the deadline drew closer. So I decided to recommend a different approach, where the whole CI system would be cloned to the local customer premises and once everything was setup and the CI system was functioning, then some of the servers would be relocated to the private cloud or where ever, the customer wants locate them.

After my assignment I have not been involved in the relocation project, but the actual relocation was six months late from planned. The recommendation about first

cloning the CI system locally has been followed and the relocation itself went quite well considering the complexity of the system. There have been some minor mishaps but nothing that has thwarted the usage of the CI system. The feedback I received from the customer was quite good and the primary goal was achieved. I got also good feedback about the non-technical issues reported and corrective measures has been started to fix those.

During this study I had a chance to work with numerous professionals and I learned a great deal about the Continuous Integration infrastructure and modern R&D methods. I learned plenty about research process and it came very clear to me, how important the planning and scoping phase is. When doing the next research I hope I can avoid the common pitfalls.

References

Berg, A. 2012. *Jenkins Continuous Integration Cookbook*. Birmingham: Packt Publishing Ltd.

Code Composer Studio (CCS) Integrated Development Environment (IDE). Page on the Texas Instrument home page. Accessed on 26 April 2015. Retrieved from http://www.ti.com/tool/CcStudio

Coghlan, D. & Brannick, T. 2014. *Doing Action Research in Your Own Organization*. 4th ed. London: SAGE

Conklin, A. & White, G. 2014. *CompTIA Security+™ All-in-One Exam Guide*. 4th ed. USA: McGraw-Hill Education

Fowler, M. 2006. *Continuous Integration*. Accessed on 27 March 2016. Retrieved from http://www.martinfowler.com/articles/continuousIntegration.html

Gilmore, T., Krantz J. & Ramirez, R. 1986. *Action Based Modes of Inquiry and The Host-Researcher Relationship*. Consultation, 6, 3, 160-176

GIT SYNC implementation. Page on the customer intranet. Accessed on 24 March 2015. Retrieved from customer intranet (Internal only)

Gitorious. Page on the Gitorious Home Page. Accessed on 24 March 2015. Retrieved from https://gitorious.org/gitorious/pages/Home

Grance, T., Hash, J., Peck S., Smith, J. & Korow-Diks K. 2002. *Security Guide for Interconnecting Information Technology Systems*. National Institute of Standards and Technology (NIST) – Tecnology Administration U.S. Department of Commerce

Information Security Risk Management. Page on the customer intranet. Accessed on 29 April 2015. Retrieved from the customer intranet (Internal use only)

IT Security Exemptions. Page on the customer intranet. Accessed on 29 April 2015. Retrieved from the customer intranet (Internal use only)

Kamel, M., Laborde, R., Barrere, F. & Benzekri A. 2009. *SecMaLET: a tool for establishing the chain of trust within a Virtual Enterprise.* 2009 International Conference on Network and Service Security, N2S '09

Kamel, M., Laborde, R., Benzekri, A. & Barrere F. 2008. *A best practices-oriented approach for establishing trust chains within Virtual Organisations*. Proceedings – IEEE International Enterprise Distributed Object Computing Workshop, EDOC, 167-174

Kim, D., Solomon M. 2014. *Fundamentals of Information Systems Security*. 2nd ed. Burlington: Jones & Barnett Learning

Kokemuller, N. *Organization Vs. Enterprise*. Page on the Houston Chronicle website. Accessed on 3 April 2016. Retrieved from http://smallbusiness.chron.com/organization-vs-enterprise-68475.html

Leased Line vs VPN – Which Technology Is Right For YOUR Business? Page on the hSo's website. Accessed on 2 April 2016. Retrieved from http://www.hso.co.uk/leased-lines/leased-lines/leased-line-vs-vpn/

MacIsaac, D. 1996. *An Introduction to Action Research*. Page on the Physics Education Server at Buffalo State Gollege. Accessed on 10 April 2016. Retrieved from http://physicsed.buffalostate.edu/danowner/actionrsch.html

O'Brien, R. 1998. *An Overview of the Methodological Approach of Action Research*. Accessed on 10 April 2016. Retrieved from http://www.web.net/~robrien/papers/arfinal.html

OpenTTCN Tester 2012 Tour. 2014. Page on the OpenTTCN home page. Accessed on 26 April 2015. Retrieved from http://www.openttcn.com/

Program Security Description ver 2.0. Accessed on 25 March 2015. Retrieved from the project team site (Internal use only)

Reid, J. 2003. *Secure Shell in the Enterprise*. Santa Clara: Myrna Rivera

Roşu, S. & Drăgoi, G. 2011. *VPN Solutions and Network Monitoring to Support Virtual Teams Work in Virtual Enterprises.* ComSIS, 8, 1

Rouse, M. & Madden, J. 2011. *Definition: Desktop Virtualization*. Accessed on 02 April 2016. Retrieved from http://searchvirtualdesktop.techtarget.com/definition/desktop-virtualization

Service Catalog – VPN. Accessed on 25 March 2015. Retrieved from the customer intranet (Internal use only)

Shimonski, R. & Alpern, N. 2009. *CompTIA Network+ Certification Study Guide*. 2nd ed. Burlington: Syngress Publishing, Inc.

Smith, G., Watson, K., Baker, W. & Pokorski II, J. 2007. *A critical balance: collaboration and security in the IT-enabled supply chain*. International Jounal of Production Research, 45, 11, 2595 – 2613

SpiraTest Features. Page on the Inflectra home page. Accessed on 23 March 2015. Retrieved from https://www.inflectra.com/SpiraTest/Highlights.aspx

Virtual Desktop User Guide. 2014. Page on the customer intranet. Accessed on 23 March 2015. Retrieved from the customer intranet (Internal use only)

VPN Client Access User Guide Hardtoken, 2011. Page on the customer intranet. Accessed on 23 March 2015. Retrieved from the customer intranet (Internal use only)

Wikipedia. 2015a. *Beyond Compare*. Accessed on 26 April 2015. Retrieved from http://en.wikipedia.org/wiki/Beyond_Compare

Wikipedia. 2015b. *Data mart*. Accessed on 14 April 2015. Retrieved from http://en.wikipedia.org/wiki/Data_mart

Wikipedia. 2015c. *Gerrit (software)*. Accessed on 24 March 2015. Retrieved from http://en.wikipedia.org/wiki/Gerrit_%28software%29

Wikipedia. 2015d. *Jenkins (software)*. Accessed on 23 March 2015. Retrieved from http://en.wikipedia.org/wiki/Jenkins_%28software%29

Wikipedia. 2015e. *Jira*. Accessed on 23 March 2015. Retrieved from http://en.wikipedia.org/wiki/JIRA

Wikipedia. 2015f. *Nagios*. Accessed on 29 April 2015 Retrieved from http://en.wikipedia.org/wiki/Nagios

Wikipedia. 2015g. *OpenGrok*. Accessed on 14 April 2015. Retrieved from http://en.wikipedia.org/wiki/OpenGrok

Wikipedia. 2016a, *Leased Line*. Accessed on 27 March 2016. Retrieved from https://en.wikipedia.org/wiki/Leased_line

Wikipedia. 2016b, *Virtual Enterprise*. Accessed on 26 March 2016. Retrieved from https://en.wikipedia.org/wiki/Virtual_enterprise

Wikipedia. 2016c. *Virtual Organization*. Accessed on 26 March 2016. Retrieved from https://en.wikipedia.org/wiki/Virtual_organization

Yan, L. 2011. *Development and Application of Desktop Virtualization Technology*. 2011 IEEE 3rd International Conference on Communication Software and Networks, ICCSN 2011, 326-329

Zanjireh, M., Kargarnejad, A. & Tayebi, M. 2006. *Virtual Enterprise Security: Importance, Challenges, and Solutions.* Proceedings of the 6th WSEAS International Conference on Applied Computer Science, 537-542