UNIVERSITY OF OSLO
Department of Informatics

Investigating Configuration Management Tools Usage in Large Infrastructure

Master Thesis

Md Jahidur Rahman

May 23, 2012

Investigating Configuration Management Tools Usage in Large Infrastructure

Master Thesis

Md Jahidur Rahman

May 23, 2012

Abstract

The large variety of configuration management (CM) tools available makes it difficult for the customers to select the appropriate one for their needs. Thus this research investigated the users' perception of CM tools in order to gain information useful for customers and CM tool developers. In total 72 system administrators were sampled and qualitative data was collected through structured questionnaires. Data was analyzed by Analytical Hierarchy Process (AHP) to find the best CM tool according to selected criteria. The most desired deployment properties were installability, configurability, scalability and stability and the most appreciated specification management properties were language, access control, monitoring and testing properties. Another important factor was whether the CM tool vendors provided good customer support. However, on the basis of people's perception CFEngine was the best tool to use in large infrastructure.

Acknowledgements

This master thesis is the final work at the University of Oslo, which is a part of the Network and System Administration master degree program, in collaboration with the University College of Oslo and Akershus.

At first I want to give thanks to God for giving me strength of mind and energy to accomplish this Master thesis.

I want to give thanks to my teacher and supervisor Mr. Ismail Hassan for encouraging and supporting me through the period of this thesis.

I would like to give thanks to Hårek Haugerud, Tore Møller, Kristen Ribu and Æleen Frisch for all kind of support through the period of this Master degree.

I want to give thanks to all of my classmates for all kinds of help and support during the last two years.

I also want to give thanks to all the respondents for giving their valuable time.

Last of all, I would like give my heartiest thank to my parents. Only due to them I am here today.

Md Jahidur Rahman

Dated: 23/05/2012

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The challenges in in maintaining large IT systems have increased with increasing number of services, nodes and the interconnections between the nodes. Thus the interest in Configuration Management (CM), which is a procedure for developing and maintaining the product's reliability, stability and various attributes as well as overall performance, plan and operational information maintaining different services [18][20], is growing [14]. However, according to the Enterprise Management Associate, about 60% of IT related problems are due to misconfiguration of the system [18]. The misconficuration of a file or machine is often caused by the system administrator Wei-Zheng and may reduce the performance of the machine [4][5]. or worse, making the system accessible to outsiders thus causing security problems. In addition, only a small number of machines are manageable by manual configuration poul-syconf, thus automatic configuration is more efficient for larger systems.

A Configuration Management tool is an automating configuration of software, which keeps track of folders, files software packages, services, IT equipment in the LAN and keeps them updated. By a good configuration management tool, the machine and its failure recovery may work quicker than when manual configuration is used [2]. CM tools are used to set up guidelines for all the machines in a network, to set up and maintain the stability, dependability and performance of the IT system. It may also assure that each machine follows the company's policies, principles and fulfills the company's business goals [1]. Advantages of CM tools compared to manual configuration is that the system will be configured in the same way independent of the administrator, the configuration will be according to the company's security policy, it is more cost and time effective, the configuration process is documented automatically, so in sum, the system may be more reliable and more easy to manage [19].

Thus the need to use CM tools in the IT field is growing [3]. There are many different CM tools available, so it is challenging for the users to select the appropriate tool suiting their needs [7]. Thus there is a need for research on the various attributes of CM tools in order to be able to compare these. The present research will therefore investigate customer's perception of various attributes of CM tools. This information will be useful for customers to better find the CM tool suiting their needs and for CM tool developers to see what the customers perceive as important and which challenges the customers face with the various CM tools so the CM tool developers may use this information to improve the CM tools to better suit the customers' needs.

Thesis goal and objectives
The goal of this research was to investigate peoples perception about different CM tools used by users. To accomplish the goal the following objectives were chosen:

- To investigate the user friendliness (easy to learn and operate) of the selected CM tools
- To study the CM tools efficiency in solving problems
- To investigate how the CM tool fit the users needs

Chapter 2

Background

In this section Configuration Management, System Configuration, Configuration Management (CM) tools and software quality will be discussed

2.1 Configuration Management:

Configuration management (CM) in engineering field has been using for several years. A high-quality configuration management policy or plan facilitates the improvement of group work successfully. It is also an efficient supervision for Information Technology System. Software configuration management is a regulation for managing the development of software system. A configuration management outcome is conditional to the needs of an organization and how it characterizes configuration management. Software configuration management is a discipline of managing and organizing the progression of software systems [20]. The standard definition of configuration management taken from IEEE standards 729-1983 [21] emphasizes on the following features of configuration management 2.1.
Identification: an identification scheme reflects the structure of the product, identifies components and their type, making them unique and accessible in some form.

Control: controlling the release of a product and changes to it throughout the lifecycle by having controls in place that ensure consistent software via the creation of a baseline product.

Status Accounting: recording and reporting the status of components and change requests, and gathering vital statistics about components in the product.

Audit and review: validating the completeness of a product and maintaining consistency among the components by ensuring that the product is a well-defined collection of components.

There are a few CM systems, which does not follow the above definition. To detain this additional functionality, it is advisable to expand the definition of configuration management as provided by the IEEE standard. These include [24]:

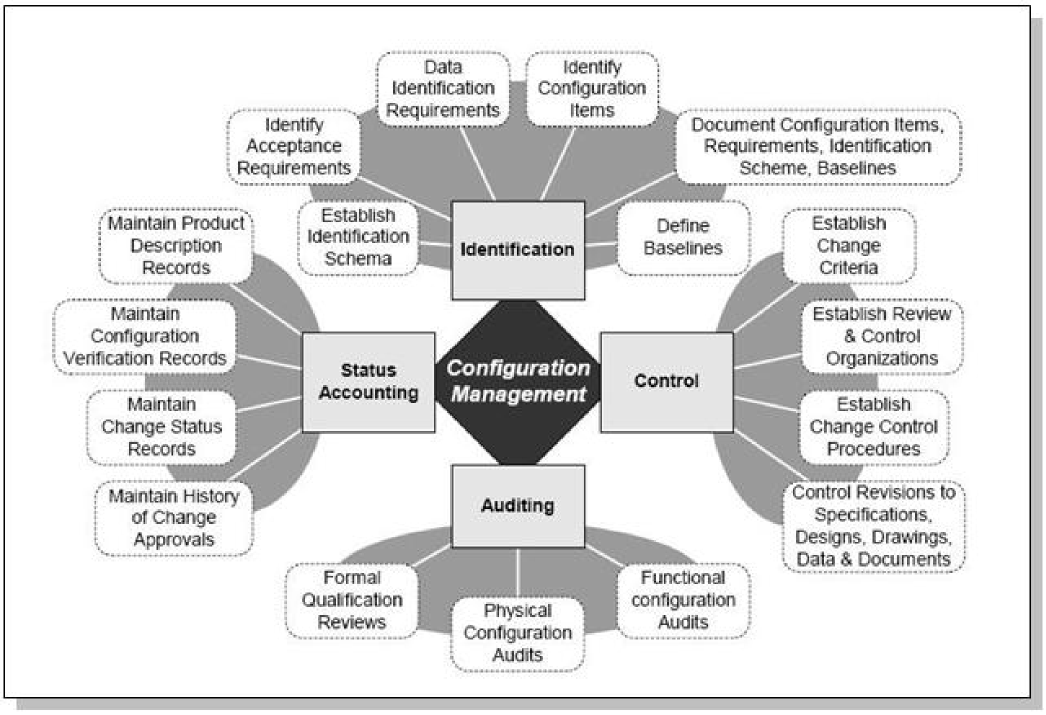

Configuration Management
Identification
Control
Auditing
Status Accounting
Data Identification Requirements
Identify Configuration Items
Identify Acceptance Requirements
Document Configuration Items, Requirements, Identification Scheme, Baselines
Establish Identification Schema
Define Baselines
Maintain Product Description Records
Maintain Configuration Verification Records
Maintain Change Status Records
Maintain History of Change Approvals
Establish Change Criteria
Establish Review & Control Organizations
Establish Change Control Procedures
Control Revisions to Specifications, Designs, Drawings, Data & Documents
Formal Qualification Reviews
Physical Configuration Audits
Functional configuration Audits

Figure 2.1: Major Functions of Configuration Management
[31]

Manufacture: managing the construction and building of the product in an optimal manner.

Process management: ensuring the carrying out of the organization's procedures, policies and lifecycle model.

Team work: controlling the work and interactions between multiple users on a product.

In sum, the configuration management abilities provided by present or running systems encompass configuration identification, configuration control, configuration status accounting, configuration verification and audit, manufacture, process management and team work [24].

Benefits of Configuration Management

Now a day's commercial networks and architectures are growing day by day, as a result the complexity is also growing with time. According to the statement published by Enterprise Management Associate, about sixty percent of IT related problem is caused due to the misconfiguration of the system. With the purpose of maintaining different services, managing hardware and software, monitoring entire infrastructure; configuration management is considering as a critical and important issue [18].

In general, Configuration Management is mainly used for technical and contractual purpose. Technically, the reason of using configuration management is to fix problems that are related to the development of software product. Those problems mainly focus on the lack of control over the software and becoming familiar with all the component of the software. Contractually, configuration management should be exercised by service provider or suppliers or developers developing software for organization. That means, certain standard should be maintained and it will help to identify the behavior of the product, how to develop and maintain it. The main objectives of using configuration management are to make sure the integrity of software and make its development easily manageable [17]. The Configuration Management is filling through every part- it influences the user's organization, the users of the Configuration Management systems, the software development environment, the quality of the software product and the software process model. The software development environment is influenced by the Configuration Management on the subject of tool combination and functionality.The Configuration Management influences the software process model and the users of the Configuration Management potentialities in terms of implementing rules and methods, approaching the users to do their tasks and it maintain a history of how the task was completed. Configuration Management influences the value of the product in the sense of how timely the product was developed and maintained. The client's organization is also affected by the Configuration Management solution because most of the organization desire that the Configuration Management solution will run internationally, all over the whole organization [22].
Configuration management is the practice of deploying, managing and keeping the latest record of all the nodes of an entire infrastructure. Say for example; in case of software, this may contain versions, updates and patches. On the other hand; in case of hardware and server, this can include the exact location and IP address of every nodes [18].
Some benefits of Configuration Management (CM)[18]:

- CM gives the opportunity of better control over all the nodes through visibility and tracking
- CM helps in advanced to detect and correct misconfiguration
- CM gives the opportunity of better asset maintenance
- CM facilitates to analysis the impact of any modification to hardware, software, firmware, documentation, testing procedures, etc
- Enhanced understanding, supervision and managing of complicated system and infrastructure

2.2 System Configuration

System configuration is the method or approach to set up a system, or the combination of different elements to make up the system 2.2. System configuration can be either hardware or software or both. When a new hardware or software is installed, some configuration is needed, that means in case of hardware say for example need to set a variety of switches and jumpers. One the other hand different values and parameters are needed to configure software [23].

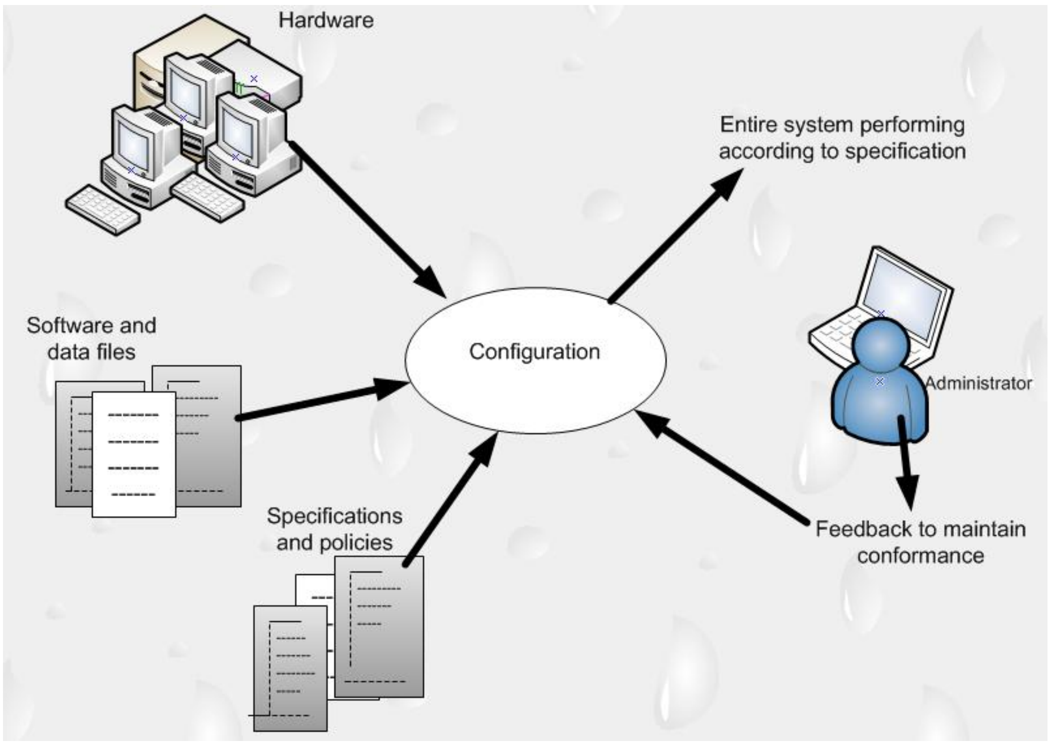

Hardware
Entire system performing according to specification
Software and data files
Configuration
Administrator
Specifications and policies
Feedback to maintain conformance

Figure 2.2: Basic system configuration task

[3]

The fundamental system configuration dilemma 2.3 is quite simple as below[14] :

- Starting with:
  - A large number of diverse machines with blank disks
  - A repository that contains all required software packages and files
  - A plan and design of all the functions, so that overall system can perform according to the proposed planned
- Install the required software and configure all the machines accordingly to make available the necessary functionality. This generally deal the internal infrastructure, for example DNS, NIS, NFS, DHCP, LDAP services.
- Configure it again when the essential service specification differs or alters
- Configure all machines again whenever the environment changes, it will help to maintain conformance with the arrangement or specification.

In general, the task of configuring a machine means editing the configuration file or providing the required information using Application Programming Interface (API) or Graphical User Interface (GUI). To configure the overall machines or system according to the guideline and specification it is very necessary to choose an appropriate configuration for every service one every machine but in practice it is difficult. To solve this difficulties a configuration system should have a model that can maintain a correlation, directed by the configuration of each machines.

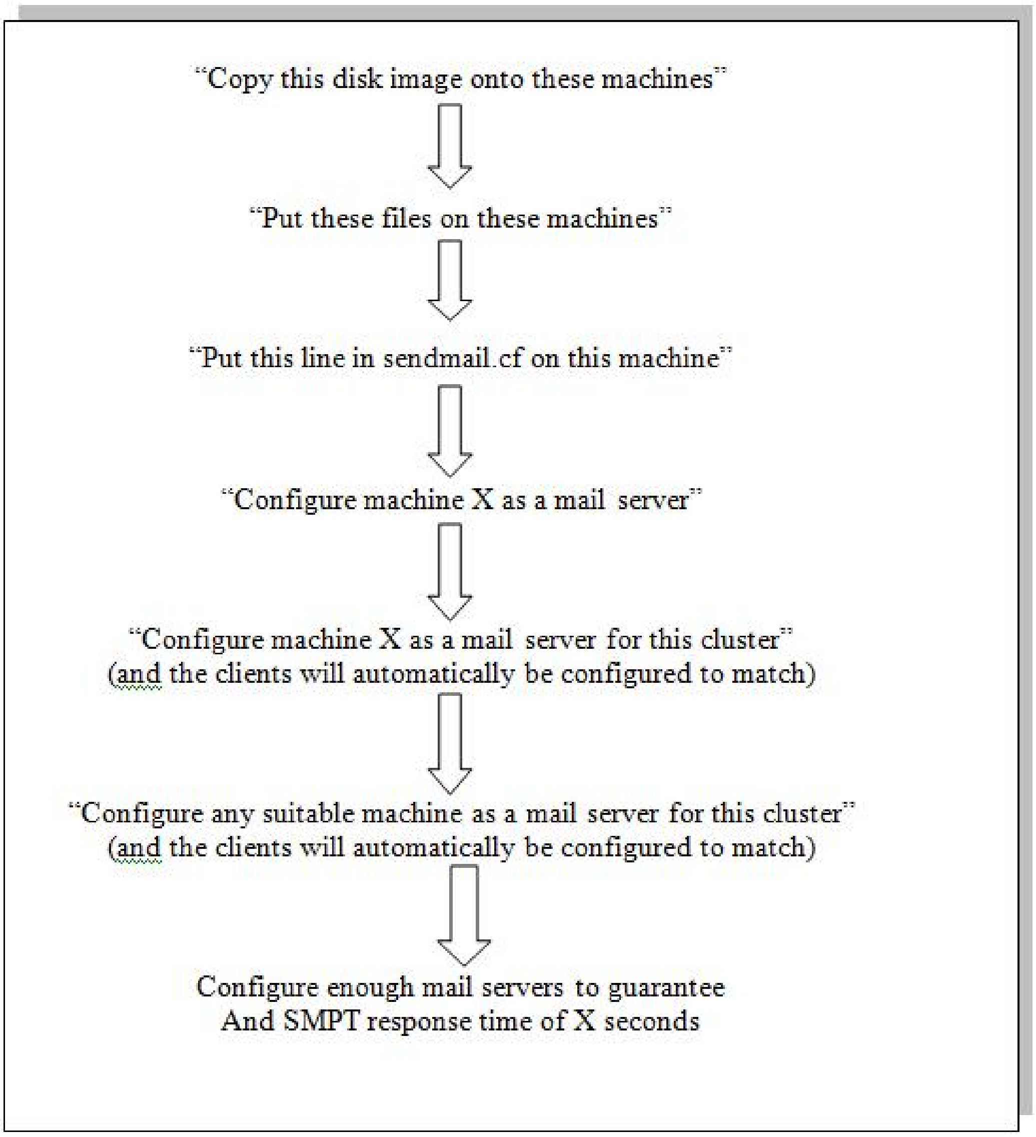

"Copy this disk image onto these machines"
"Put these files on these machines"
"Put this line in sendmail.cf on this machine"
"Configure machine X as a mail server"
"Configure machine X as a mail server for this cluster"
(and the clients will automatically be configured to match)
"Configure any suitable machine as a mail server for this cluster"
(and the clients will automatically be configured to match)
Configure enough mail servers to guarantee
And SMPT response time of X seconds

Figure 2.3: Levels of configuration specification

[3]

2.3 CM Tools

Configuration management tools are the tools and services that are used to help the sysadmin groups to deploy, administer, supervise and maintain the configuration items of a system [9].It offers and gives different types of automation and representational model, but all of them have a common goal that is to assist the system administrator by deploying and configuring the system the during the system configuration process [11]. There are over a dozen different configuration management tools are actively using to maintain large IT infrastructure such as CFEngine, Puppet, Chef, Bcfg2, Lcfg, BMC Bladelogic Server Automation Suit, CA Network and Systems Managements, IBM Tivoli System Automation for Multiplatforms, Microsoft Server Center Configuration Manager, HP server Automation System, Netomata Config Generator and etc. As so many configuration management tools are actively using to maintain large IT infrastructures,this so many options may confuse sysadmins. This section will give some basic or general characteristics of different CM tools [10].

CFEngine:

CFEngine is a very popular open source configuration management tool, which was developed by Mark Burgess in 1993. Its main role is to provide automatic installation, configuration, monitoring and maintenance support of a large-scale infrastructure. It formulates the simple administrative work automated and difficult task simpler [15]. The main features of CFEngine are as follows [16]:

- Operating System based independent language
- To configure server and agent we need not to maintain separate configuration file because in CFEngine the configuration procedure is centralized
- Policy server is also maintained centrally
- It also supports decentralization to facilitate the agent to run separately
- For remote establishment general access control support role based access control
- Powerful pattern matching and expression features. Support text matching
- Enhanced array and list handling
- Enhanced standard package management

General Characteristics:
CFEngine is based on C language. It uses a declarative DSL for their input specification. CFEngine is a completely decentralized framework, with pull based client-server system. It uses command line interface to manage the system. CFEngine uses the idea of classes to group configuration. To construct hierarchies, classes contain other classes. Using expressions it assigns classes statically or conditionally. To manage a subsystem and device CFEngine uses bundles to allocate reusable configuration specification. For modeling relations between instances CFEngine supports one-to-one, one-to-many and many-to-many relations. For workflow mechanism CFEngine support coordination of distributed changes and it uses it uses strongly distributed management system to manage each node. It support almost all kind of famous platform like *BSD, AIX, HP-UX, Linux, Mac OS X, Solaris and Windows. It uses dry run mode and integrated monitoring system to support testing specification and monitoring the entire infrastructure respectively. For versioning support CFEngine uses a textual input and this textual input is managed using an external repository. To generate documentation it uses structured comments. For integration with environment CFEngine can determine runtime characteristics of supervised nodes. CFEngine supports conflict management by detecting modality conflict. To limit the access of the user and administrator CFEngine use path based access control. To support valued customers

CFEngine provide good documentation and tutorials. It also offer training for the customers or users [7].
Strength:
CFEngine is able to run on multiplatform. It is a very generic configuration management tool. It is the mostly used configuration management tool. CFEngine is capable to run in a degrade situation. Cfengine runs very fast .It uses very few memory of the system; only 30MB.It has no dependency. It doesn't dependent on external interpreter. It has high-quality security record. When problem arise it also help us to decide how to fix the problem. .Another strength of CFEngine is it's support; it has good documentation, tutorial and user community [10].
Weakness:
Difficult to begin because it is very vast and there is a lot to learn and it's syntax [10].

Puppet:

Puppet is an open source configuration management tool to manage both unix-like and windows platform based IT infrastructure centrally and automatically. It was established by Luke Kanies in 2005 and released under the GNU public license.For installation and configuration Puppet uses a client-server deployment, where agent gets configuration and installation instruction from a central puppet master [8].
General Characteristics:
Puppet is written using Ruby language. It uses a declarative and imperative Rubby DSL (Domain Specific Language) for its input specification. It uses both graphical user interface and command line interface to manage the system. Like CFEngine Puppet also uses the idea of classes to group configuration. To construct hierarchies, classes contain other classes and it allots classes dynamically with the help of external tools. To manage a subsystem and device Puppet uses modules to allocate reusable configuration specification to manage certain nodes or subsystems. For modeling of relations between instances Puppet can define one-to-one and one-to-many relations but it does not support many-to-many. To support workflow Puppet uses coordination of distributed changes and this is done by exporting and collecting resources between managed nodes. Like CFEngine it support almost all kind of famous platform like *BSD, AIX, HP-UX, Linux, Mac OS X, Solaris and Windows. To support for testing specification Puppet use dry-run mode and it also support several environments like testing, staging and production.For versioning support like CFEngine, Puppet also uses a textual input and this textual input is managed using an external repository. Puppet can create documentation from the comments integrated with the source code and its specification may contain free from comments. To generate documentation it uses structured comments. For integration with environment Puppet is able to determine runtime characteristics of supervised nodes. It also supports conflict management by detecting modality conflict. To limit the access of the user and administrator like CFEngine, Puppet also use path based access control. Puppet also provides good documentation and tutorials to give support its customers. It also offer training for the customers or users [7].
Strengths:

- Very big user community (On the Puppet mailing list there are over 2000 users)[10]

- It has a high-level model of system concept [10]

Weaknesses:

- Puppet server fetched bottleneck problem. [10]
- Its execution of order is not deterministic. [10]

	Language	License	Authentication	Encryption	Release
CFEngine	C	GPL, COSL	Yes	Yes	1993
Puppet	Ruby	GPL	Yes	Yes	2005
Bcfg2	Python	Bcfg2 License	Yes	Yes	2004
Chef	Ruby	Apache	Yes	Yes	2009
Lcfg	Perl	GPL	Partial	Partial	1994

Table 2.1: Basic properties of different CM tools

[12]

Chef:

Chef is an open source configuration management tool written by Opscode. It helps to bring the advantages of configuration management to our whole IT infrastructure. Chef is written In Ruby and it runs under client/server model or on a combined configuration known as "chef-solo".
General Characteristics:
As a language Chef uses ruby DSL. Like Puppet Chef also use both graphical and command line user interface. For grouping Chef define static group and groups based on queries. To manage a subsystem and device Chef uses a cookbook for allocating reusable configuration specification to manage certain nodes or subsystems. For modeling of relations between instances Chef uses many-to-many dependency. Chef does not provide any support for workflow. It runs on *BSD, Linux, Mac OS X, Solaris and windows. Like Puppet, Chef also use dry-run mode for testing specification and also support several environments like testing, staging and production. For versioning support Chef uses a textual input to make its own configuration specification. A cookbook for version information is also maintained by the Chef central server. Like Puppet, Chef also can create documentation from the comments integrated with the source code and its specification may contain free from comments. Like CFEngine its access control system is path-based. It provides wide-ranging reference documentation and tutorials, it also gives commercial support to its userscite [7].

Strengths:[10]

- Involuntary provisioning and configuration of fresh nodes
- Multi-node orchestration
- Reusable of policy cookbook
- Cloud incorporation

Weaknesses:[10] Attributes has several levels of preference and this is overwhelming

Tool	Platform support
CFEngine	*BSD, AIX, Linux, Mac OS X, HP-UX, Windows and Solaris
Puppet	*BSD, AIX, Linux, Mac OS X, Solaris
Bcfg2	*BSD, AIX, Linux, Mac OS X and Solaris
Chef	*BSD,Linux, Mac OS X, Solaris and Windows
Lcfg	Linux
BMC Bladelogic	AIX, HP-UX,Linux, Network equipment,Solaris, Windows
CA Network and Systems Management	AIX, HP-UX,Linux,Solaris Mac OS X, Windows Network equipment
IBM Tivoli System Automation for Multiplatforms	AIX,Linux,Solaris and Windows
Microsoft Server Center Configuration Manager	Windows
HP Server Automation System	AIX,HP-UX,Linux Network equipment Windows and Solaris
Netomata Config Generator	Network equipment

Table 2.2: The platforms that each tool supports

[7]

Bcfg2:

The configuration management tool Bcfg2 helps sysadmin people construct or establish a reliable, dependable, reproducible and demonstrable or provable explanation of their environment. It facilitates system administrators by providing virtualization and reporting in their daily administrative works. It was created by Narayan Desai and developed by the division of Mathematics and Computer of Argonne National Laboratory [13].
General Characteristics:
Bcfg2 is written using Python language. It uses a declarative DSL for its input specification. To manage the system it only uses the command-line interface but doesn't support for graphical interface. As a grouping mechanism Bcfg2 use static grouping and hierarchies of group but it has no support for configuration modules. Moreover, Bcfg2 does not support for modeling of relation in a configuration specification. It runs on *BSD, AIX, Linux, Mac OS X and Solaris platform but not on windows. Dry run mode is used to support for testing specifications. For versioning control it uses a textual input to create its own configuration specification. However, it presents wide-ranging reference documentation and tutorials but it does not provide any commercial support [7].
Strengths: [10]

- Reporting system
- Debugging

Weaknesses:

- Documentation is not well organized and reach
- Not easy to share the policies between different sites

	CFEngine	Puppet	Chef	Bcfg2	HP	IBM
Language type	Declarative	Declarative Imperative	Imperative	Declarative	Imperative	Declarative
User interface	Command line	Command line,GUI	Command line,GUI	Command line	GUI	Command line,GUI
Type of grouping	Static, Querybased, Hierarchical	Static, Querybased, Hierarchical	Static, Querybased	Static, Hierarchical	Static	Static
Modeling of relations	many-to-many, one-to-many, one-to-one	one-to-many, one-to-one	many-to-many	-	-	many-to-many, one-to-many, one-to-one
Scalability	>10000	1000-10000	unknown	<1000	unknown	unknown
Workflow	coordination of distributed changes	coordination of distributed changes	-	coordination of distributed changes		coordination of distributed changes
Translation agent	strongly distributed management	centralized management	centralized management	centralized management	centralized management	centralized management
Ease of use	medium	medium	hard	hard	easy	easy
Versioning support	yes	yes	yes	yes	-	yes
Access control	path based	path based	path based	path based	hierarchical	hierarchical
Support	document-ation, reference, tutorial, training	document-ation, reference, tutorial, training	reference, tutorial, training	document-ation, reference, tutorial	training	document-ation, reference, tutorial, training

Table 2.3: General characteristics different CM tools

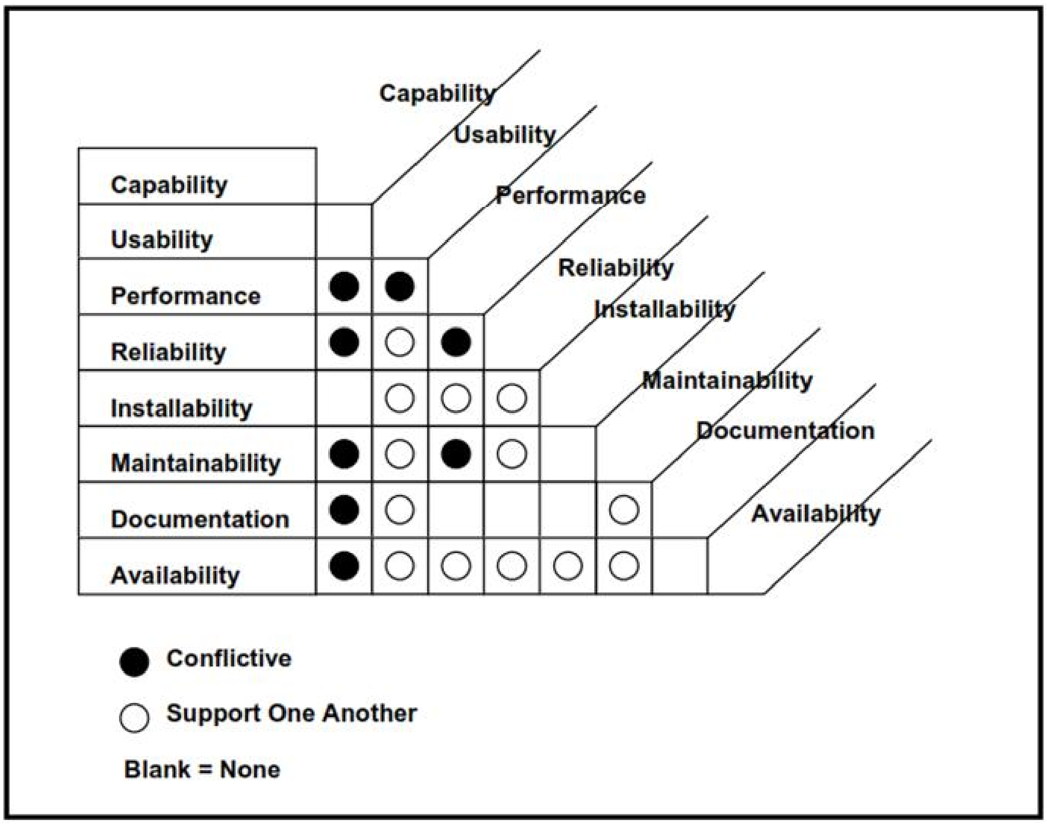
[7]

2.4 Software Quality

In the book *"Quality is free: the art of making quality certain"* [29], Philip B. Crosby mentioned:
"The first erroneous assumption is that quality means goodness, or luxury or shininess. The word quality is often used to signify the relative worth of something in such phrases as "good quality", "bad quality" and "quality of life" - which means different things to each and every person. As follows quality must be defined as "conformance to requirements" if we are to manage it. Consequently, the nonconformance detected is the absence of quality, quality problems become nonconformance problems, and quality becomes definable"
By our experience and practice we know that most of the software generally has "bugs". For a software developer it is a very big challenge to develop good software without any bug. We can define the software quality by using two ways: Reliability and Defect rate. Reliability means the total number of breakdown or stoppage per n hours or the possibility of a continuous operation in a given time without any failure. Reliability measures the level of threat and the probability of application failures. The purpose for inspecting and examining Reliability is to decrease and avoid application downtime that openly affects end users. And by the defect rate it means the number of mistakes or faults found with respect to per hundreds or thousands of lines in a given source code.By means of customer satisfaction survey, customer satisfaction level is calculated in percentage that means how much customer in percentage satisfied or not satisfied. Usually for this kind of survey blind survey technique is used because it will help to decrease bias. For example, IBM examines satisfaction with its own software products in levels of CUPRIMDSO (capability, usability, reliability, installability, performance, documentation, maintainability, service, and overall). HP's center of attention on FURPS (functionality, usability, reliability, performance, and serviceability) [25].To achieve customer satisfaction increasingly, the quality features have to be taken into account when developing the software; though these quality features are not always suitable or compatible with each other. Say for example, when the functional complexity of software is higher it makes harder to achieve maintainability. So those points are evaluated by different customers in different ways .For instance, performance and reliability might be considered as the most important quality for large users with complicated network and real-time processing. On the other hand those customers who are using standalone system and simple operation may be considered usability, installability and documentation as more important attributes. The figure 2.4shows a probable relationship between different types quality attributes. Depending on the customer types and applications, some relationships are supportive, some of them are negative and rest of them are not clear [25]. Usually software developer companies assess their customer's satisfaction as a quality index, for example, using the levels of CUPRIMDSO IBM ranks their software products [27]:

- *Capability:*
  It refers to the software satisfying its useful requirements
- *Usability:*
  It refers to the essential attempted to learn, and manage the software

- *Performance:*
  It refers to the software performance and resource utilization
- *Reliability:*
  It refers to software fault tolerance and recoverability
- *Instalability:*
  It refers to the necessary effort to install and implement
- *Maintainability:*
  It refers to the necessary effort to regulate the software
- *Documentation:*
  It refers to the coverage and accessibility of the software documentation
- *Availability:*
  It refers to how easily we can access.



	Capability	Usability	Performance	Reliability	Installability	Maintainability	Documentation	Availability
Capability								
Usability								
Performance	●	●						
Reliability	●	○	●					
Installability		○	○	○				
Maintainability	●	○	●	○				
Documentation	●	○				○		
Availability	●	○	○	○	○	○		

● Conflictive

○ Support One Another

Blank = None

Figure 2.4: Interrelationship of Software Attributes:A CUPRIMDA Example [25]

Different software has different type of users, for this reason it will be heard to set objective for the quality attributes. As referred to paper [26], about fifteen percent of all software faults are requirement errors. A software will always be considered as a bad quality software if the development process does not consider and emphasize on requirements of the quality. Process quality is also another important thing for software quality in contrast with end-product quality. The development process is a composed of different states. Every state dependent's on its earlier state and every state has its own quality features which can influence the end product.

2.5 Customer focus on QoS

"Quality of Software (QoS) must be defined and measured for improvement to be achieved" [28]. A well-liked observation of quality is that it is treated as intangibly and we can talked about and judged but we cannot be computed. From a customer's point of view quality of software is the customer's evaluation about software that is used by them for their needs. Normally when customer's judge or evaluate or want to buy a software they always consider a range of variables, like the performance, reliability, satisfaction and price [28]. From customer standpoint, quality of software is recognized as independent of any quantifiable characteristics. That is, quality is described as the products or services that are capable to satisfy customer needs and desires - explicit or not describe [30].

- *Performance*
  How much pleased with the performance of the software? Is the software fully capable to meet our all needs and desire? Is it fulfilling our needs? This is the most important factor when a customer wants to evaluate software.

- *Reliability*
  Is the software consistent? Is the software performing well under a stated circumstance for a particular period of time?

- *Satisfaction*
  How enjoyable the software is to use the design? Are we fully satisfied with the performance of the software?

- *Price*
  When customers want to buy new software they always first think about the price of the software. But if they previously well informed about the software then they think do we actually want this software and do we truly have enough money for this software?

In Guasparis book *I Know It When I See* the author discusses quality in the customerscontext as follows [25]:

"Your customers are in a perfect position to tell you about Quality, because thats all theyre really buying. Theyre not buying a product. Theyre buying your assurances that their expectations for that product will be met.And you havent really got anything else to sell them but those assurances. You havent really got anything else to sell but Quality"

One of the main goals for every software developer company is to achieve complete customer satisfaction. To do this they emphasize on customers wants and needs, collects customer requirements and determining and supervising customer satisfaction. Several steps are involved for making high quality software; one of them is to ensure customer satisfaction. To develop a quality full software, the customers prerequisite and needs must collected first and analyzed, specifications to meet those necessities and requests must be fulfilled, and the software must be build up and manufactured accordingly [25].

Chapter 3

Methods

3.1 Sampling

Respondents were selected on the basis of their experience that means those who are using or maintaining IT infrastructure with the help of CM tools. In total 72 respondents from USA and Europe was sampled.

3.2 Data collection

Qualitative data were collected using structured questionnaire through a survey with 72 respondents in January and April 2012. In total 41 electronic questionnaires were sent to respondents in USA and various European countries. In addition, 31 interviews, where the questionnaire was filled out either by the respondent or the interviewer in the presence of the interviewer, were conducted in USA. Electronic survey was chosen due to time limitation and cost efficiency. All questionnaires contained the same, closed ended questions about the user's satisfaction with the configuration management tools' user friendliness, security, ability to solve problems related to configuration and maintenance and price of the management tool.

Secondary data included publications, thesis papers, books and internet sources on manual and automated configuration management.

For Bibliography EndNote X4 with numbered style was used

3.3 Data analysis

There are many decision making methods for selecting an open source software, such as Pros and cons Analysis method, Lexicographic method, Maximum and minimum method, Generalized means method and Analytic Hierarchy Process (AHP) [50]. For this research the AHP was selected as the method allows the use of multiple variables, or user preferences, to calculate the best fitted CM tool for the user. AHP is a decision making algorithm, which consider multicriteria according to hierarchal approach, determining the importance of those criteria, contrasting substitutes for every criteria and ranking the options.

It finds the best tool or software based on the indicated goals and needs [51]. The AHP method follows these steps to find the best CM tool (for detailed analysis see appendix 1) I) determines the weights for the criteria II) determines the rating for every chosen alternative for every criterion and III) computes the weighted average rating for every decision alternative. IV) select the best one which acquires the height score[52][53].

Step One: Determine the weights for the criteria The respondents allocated values to the selected criteria (Support, Deployment and Management).

Step two: Determine the rating for every chosen alternative for every criterion. The values allocated by the respondents were used in the following equation to compute pairwise comparison value matrixes for every criteria:
Equation (I)

$$\begin{array}{cccc} & Criterion1 & Criterion2 & Criterion3 \\ Criterion1 & \frac{c1}{c1} = a_{11} & \frac{c1}{c2} = a_{12} & \frac{c1}{c3} = a_{13} \\ Criterion2 & \frac{c2}{c1} = a_{21} & \frac{c2}{c2} = a_{22} & \frac{c2}{c3} = a_{23} \\ Criterion3 & \frac{c3}{c1} = a_{31} & \frac{c3}{c2} = a_{32} & \frac{c3}{c3} = a_{13} \end{array}$$

The 3rd root of every criteria were calculated and the sum of the 3rd roots was used in the following equation to calculate the priority vector by normalizing the matrices: Insert Equation (II)

$$a'_{ij} = a_{ij} \Big/ \sum_{i=1}^{n} a_{ij} \quad , i, j = 1, 2, \ldots, \mathrm{n}$$

$$\begin{array}{cccc} Objective & criterion\ 1 & criterion\ 2 & criterion\ 3 \\ criterion\ 1 & \frac{a_{11}}{\sum_{i=1, j=1}^{3} a_{ij}} & \frac{a_{12}}{\sum_{i=1, j=2}^{3} a_{ij}} & \frac{a_{13}}{\sum_{i=1, j=3}^{3} a_{ij}} \\ criterion\ 2 & \frac{a_{21}}{\sum_{i=1, j=1}^{3} a_{ij}} & \frac{a_{22}}{\sum_{i=1, j=2}^{3} a_{ij}} & \frac{a_{23}}{\sum_{i=1, j=3}^{3} a_{ij}} \\ criterion\ 3 & \frac{a_{31}}{\sum_{i=1, j=1}^{3} a_{ij}} & \frac{a_{32}}{\sum_{i=1, j=2}^{3} a_{ij}} & \frac{a_{33}}{\sum_{i=1, j=3}^{3} a_{ij}} \end{array}$$

Then the Consistency Ratio (CR) was calculated in the following four procedures: **Procedure one:** The values of each column were added and the calculated sum was multiplied by the respective weight by the following formula:

AW^T..(III)

Where A is the pairwise comparison matrix and W$\hat{T}$ is the weight vector.

Procedure two: The values in the Sum*Pv row were summarized and inserted in the 3rd root of criteria column. This is the value of the lambda-max.
Procedure three: The Consistency Index (CI) was calculated using the formula:

$$\lambda_{\max} = \frac{1}{n} \sum_{i=1}^{n} \frac{(Aw^T)_i}{(w^T)_i}$$

Procedure three: At this stage the Consistency Index (CI) is calculated using the formula:

$$CI = \frac{\lambda_{\max} - n}{n - 1}$$

Where n is the number of criteria. Here n=3.
Procedure Four: The Consistency Ratio (CR) was calculated and by dividing the Consistency Index (CI) by the Random Index (RI).
The Consistency Ratio illustrates how much consistence of the decision maker in the pairwise comparison. If CR <= 0.10 the pairwise comparison matrix is consistence enough and if CR >= 0.10, the pairwise comparison matrix is strongly consistence.

Step three: Compute the weighted average rating for every chosen option and select theone which acquires the maximum score.

Then rating for every selected option for every criterion was developed. Then the score for each CM tool was calculated to find the best CM tool.

Chapter 4

Results and Discussions

There are many configuration management tools available, and users often use the following selection criterions installability, configurability, scalability, usability, security, support and platform support. Beside these, they also consider the policies and size of their own and the manufacturer's company and the cost of the CM tool. Thus the following sections will compare the CM tools based on some of these selection criterions.

4.1 The type of configuration management tools used

The first question was what kind of Configuration Management Tool the participant use. The purpose of this question was to find out which CM tools are mostly used.At present there are many configuration management tools available in the market and every tool has their own aim, characteristics and target groups; but all of them have a common or widespread goal that is to assist in the system configuration process. When sysadmin people want to select a tool they always consider the following factors: install-ability, configure-ability, scalability, usability, support, security and platform support. Beside these, they also consider the company policy, size of the company and the cost of the tool during the selection of a tool. Based on those factors people are choosing their configuration management tools for their environment.

The most widely used CM tools were Puppet (32%) and CFEngine (30%) (Fig. 4.1)About 15% used Bcfg2, 13% used Lcfg, 8% used Chef 4% used BMC Bladelogic and 3% used IBM Tivoli.

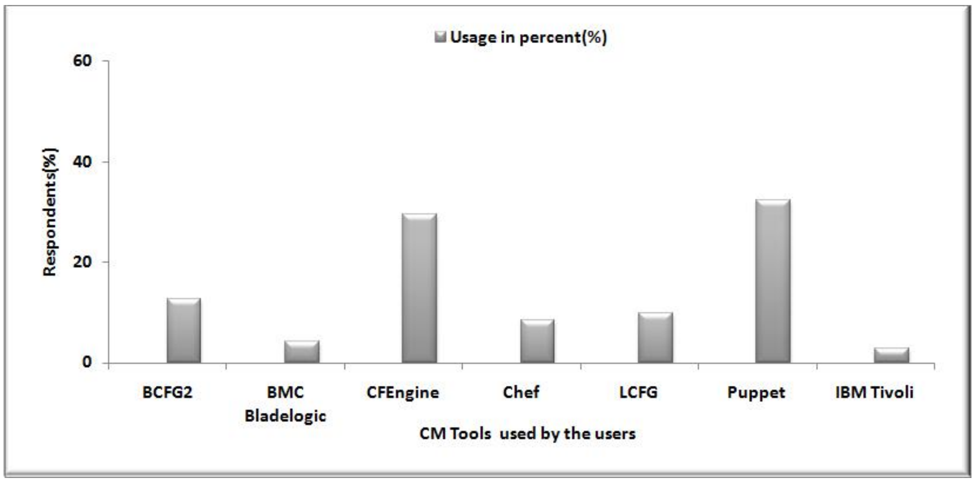


Figure 4.1: CM tools used by the users

4.2 Installability

Installability, whch is how capable the software is to be installed in a particular environment, can influence the consequence outcomes of suitability and operability. The installability of the software depends on the factors such as number of steps in the installation process, time required for installing the software, resource consumption during the installation process, ability to back out of the installation process, ability to uninstall, dependency that is additional tools and techniques to install the software successfully, installation media, policy and standard needed to install the software [32].

The figure (Fig.4.2) shows that about 90% of the CFEngine users found CFEngine difficult to install. These 90% were further analyzed and about 29% found CFEngine installation process slightly difficult and 10% found it very difficult. However, out of the total CFEngine users, 5% found it very easy and easy to install respectively.

The installation process of CFEngine is divided in three major components: 1) Server (cf-serverd) is configured to set up guidelines for the LAN, 2) Client (cf-agent) is configured to accept connections and implement instructions from the central server and 3) Scheduler (cf execd) is configured to control the implementation of the jobs and that all the nodes are running well. To make the system fully automated using CFEngine, it is needed to configure the components separately. In addition, there are many extra tools for particular tasks such as the cfkey and cfrun [16].May be due to these complex installation procedure and configuration syntax such a high percentage of the respondents found CFEngine difficult to install, and maybe those who found it easy to install were experienced users.
About half of the Puppet users found that CM tool to be easy to install, and only 14% and 36% found itdifficult orslightly difficult to install respectively and slightly difficult.
In Puppet only one machine is considered the master server and all other nodes are

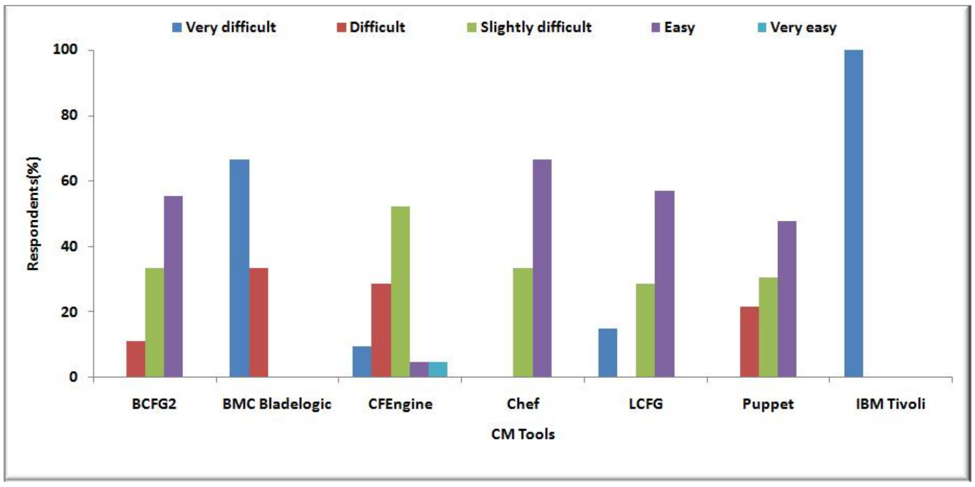


Figure 4.2: Installation difficulties of different CM tools

puppet clients. Instructions or rules are set in the Puppet master, which spreads them to the connected nodes This simple procedure makes puppet easy to install [33] and is probably the reason why Puppet was considered by most of its users to be easy to install About 67% of the Bcfg2 users found that CM tool "Easy" toinstall (Fig.4.2). This may be because Bcfg2 is written in Python and only requires python-lxml andpython-ssl. Thus thereis no need to install any supplementary scripting, programming language or complex dependencies on existing or older systems [36].

About 61% of the Chef users found CM tool "easy" to install, and only 33% found it "slightly difficult". This may be as the CM tool is easy to start and it is flexible, so the user can use API or shellcommand to manage it, and user manuals are easily available. It may be that the users who found the CM tool "slightly difficult" to install needed Ruby and many dependencies for successfull installation [34]. Above half of the LCFG (57%) and Bcfg2 users (56%) found the respective CM tools easy to install. About 15% and 29% of the LCFG users found the CM tool very difficult and slightly difficult to install 33%of the Bcfg2 users found it difficult to install (Fig.4.2).

To make the system automated the server package is installed on the Bcfg2 server and clients packages on connected machines. Maybe due to the need to also install the CM tool on the client servers made I the installation process difficult for the novice users, but easy for the experienced. Chef was considered as the easiest CM tool to install. IBM Tivoli and BMC Bladelogic were the most difficult CM tools to install as as because the installation is push based, time consuming and monitoring requires certain competency [15].

4.3 Configurability

Different CM tools have different functional units and to obtain the maximum benefit of the CM tool it is necessary to configure it appropriately as the configurations influence the system function and performance [35].

The figure 4.3 described that nearly all the CFEngine users found the CM tool difficult to configure and only 5% found it easy. Among the users findin it difficult to configure, 15% found it very difficult, 38% difficult and about 43% slightly difficult.The CFEngine has many components, each with its own function.

The basic components are Server (cf-serverd) thatsets up guidelines or rules which apply on the LAN. Client (cf-agent) that is configured to accept connections and implement instructions from the server and Scheduler (cf-execd) which controls the implementation of the jobs and that the nodes are running well. These components need to be configured Separately to make the systm fully automated. In addition there are many tools for particular tasks such as cfkey and cfrun [16]. Thus, for a novice user it is difficult and time consuming to configure CFEngine as it requires a lot of knowledge [10].The CFEngineis treated as a programming language because it has its own syntax, sothese complex configuration procedures may be to the reasons why the users found this CM tool difficult to configure. Maybe the 5% who found CFEngine easy to configure are experienced using CFEngine, and no user found it"very easy" to configure.

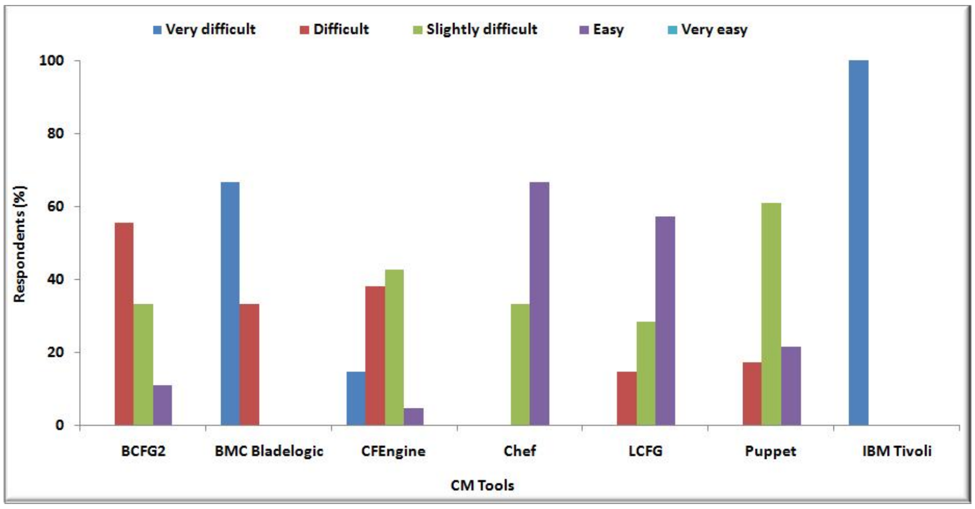


Figure 4.3: Configuration difficulties of different CM tools

About 43% of the Puppet users found the configuration procedure of Puppet to be "slightly difficult", but only 17% found it "difficult".and 22% found it "Easy". As explained under "Installability" the configured file is set in the Puppet master server, which pushes it to connected nodes. This simple procedure makes Puppet easy to configure [33]. On the other hand, Puppet is treated as a programming language because it has its own syntax, thus the users who found Puppet difficult to configure may be new in operating this CM tool.

Almost all the users of Bcfg2 found it "Difficult" (68%) and "Slightly difficult" (33%) to configure.

To install Bcfg2 the server and client are configured separately and in difference to

Puppet or CFEngine, the Bcfge2 configuration also requires configuration of folders in a hierachy [37]. The major part of the IBM Tivoli (100%) and BMC Bladelogic (67%) users found those CM tools very difficult to configure, which corresponds to the proportions finding it difficult to instal those CM tools.. The easiest configuration tool is Chef, about 67% of the Chef users considered it as easy to install.

4.4 Scalability

Scalability is how the system performance is influenced by adding or removing resources and/or load [38].The CM tool scale can be defined by testing and analyzing each tool in the deployment and scale the number of running nodes. To evaluate the CM tools, five groups were created based how many nodes they are able to operate.The CM tools managing 0-100 nodes were considered as a very small company, 100-1000 nodes as a small company, 1000-10000 nodes as medium company, 10000-100000 as large company and more than 100000 was considered as very large company.

CFEngine and Puppet were very scalable compared to the other CM tools and both tools were used in a wide range of companies from very small to very large (Fig.4.4) CFEngine and Puppet are used in small companies (19% and 39% respectively), medium companies (33% and 35% respectively), large companies (33% and 4% respectively) and very large companies (15% and 9% respectively). Bcfg2, Lcfg, Chef, IBM Tivoli and BMC Bladelogic were used for very small to medium sized companies, but 22% of the Bcfg2 users used the CM tool to manage large IT companies. According to Delaet et al.,Bcfg2 supports less than 1000 nodes, LCFG and Puppet can manage between 1000 and 10000 nodes, BMC Bladelogic and CFEngine can manage more than 10000 nodes and the scalability of Chef IBM Tivoli was not known [7].

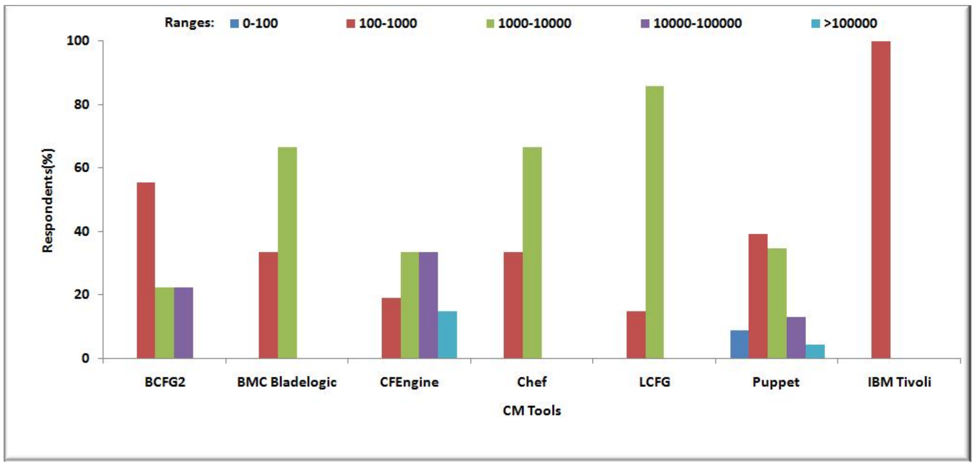


Figure 4.4: How scalable the CM tools are

4.5 Stability

Good quality software is assessed based on its stability, flexibility, portability, reliability, integrity, testability, efficiency and maintainability.Stability, which means that the software or system can run for a long time without breakdowns, is one of the most important attributes as many of the other attributes depends on this [39]. As the system environments and requirements are changing frequently, software maintenance or configurationis a continuous process. The newly configured application or software must give immediate or rapid response according to the system needs withoutcompromising with system quality [40] and A stable software can fulfill these needs.
Only 5% of the CFEngine users found their CM tool "very unstable", which may be due to misconfiguration, hardware failure or software bugs(Fig.4.5). About 43% of the CFEngine users said that CFEngine is stable and 83% of the Puppet users thought the same about the Puppet. The majority of the Bcfg2, Lcfg and Chef users found their CM tools "Very stable" (29%, 33% and 33% respectively) and "Stable" (71%,44% and 67% respectively). About 23% of the Bcfg2 users found the CM tool unstable. This could be due to bugs of the software,misconfiguration or hardware problem.

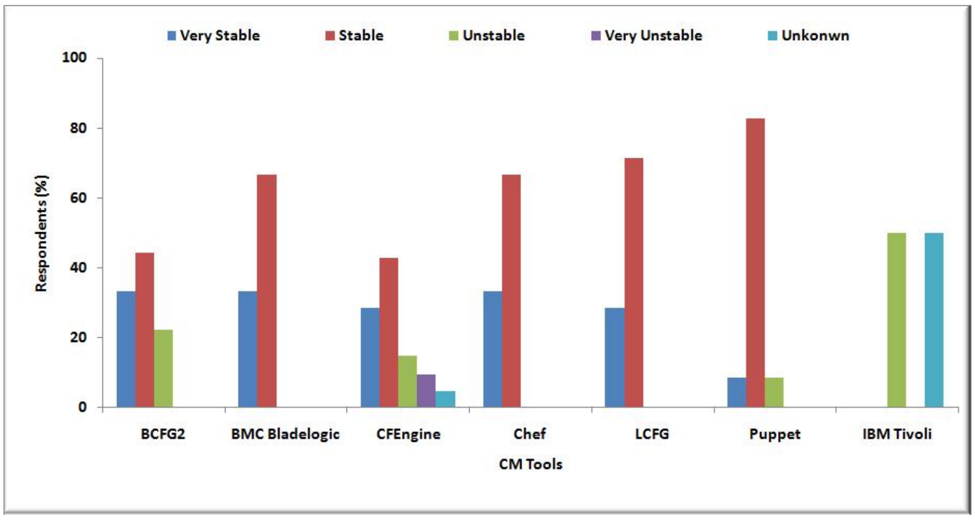


Figure 4.5: How satbel the CM tools are according to the users experience

4.6 Usability

Usability, according to ISO, refers to "the extent to which a productcan be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [41].

About 67%, 33% and 20% of the BMC Bladelogic,Lcfg and CFEngine users found their CM tools very difficult to use (Fig.4.6). In addition, most of the CFEngine and Puppet users (57% and 65% respectively) as well as 33% and 29% of the Chef and Lcfg users found their CM tool "slightly difficult" to use and no CFEngine user found itvery easy to use.. In the same way not a single user of Puppet didn't marked it as "very difficult" to use .In case of puppet 26% user said it is easy to use. No Bcfg2 or Chef users found those CM tools difficult or very difficult to use, but according to Delaet et al.[7] the XML input and the plugin system of the Bcfg2 and Chef syntax and terminology make these CM tools difficult to use.

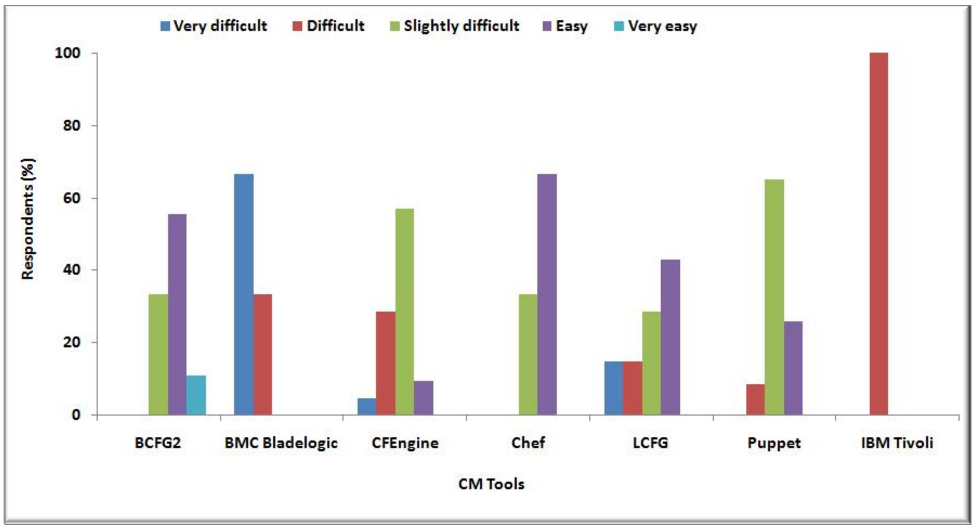


Figure 4.6: How user friendly the tools are according to the user's experience

4.7 Documentation

Documentation is a document, such as a user guide, white paper, on line help or quick reference guide, explaininghow to use, install, maintain, and assemble the product. A good document can can also help to improve, extend and update the software [42], thus it is important for sustaining the tool [7].

The users of Chef (83%), LFCD (71%) and Puppet (57%) were the most satisfied with the documents about their CM tool Fig.4.7). The first two have extensive documentation on their website and Puppet has a user manual of 30 pages [15]. Also 33% of the CFEngine users were satisfied with the documentation. The CFEngine user manual is about 500 pages [15], so the satisfied CFEngine users may be experienced system administrators who need extensive information. However, about half of the CFEngine users (48%) were dissatisfied with the documentation, which may also be to the extensive user manual [15]. BCFG2 posted its documentation on its wiki site in addition to a 30-page online document , but BMC Bladelogic did not give any support by public documentation [7].

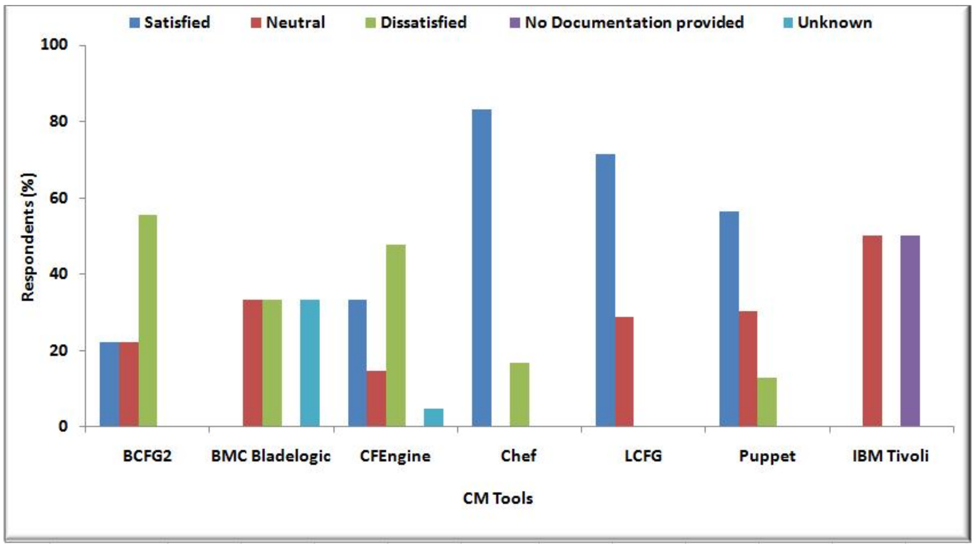


Figure 4.7: Level of satisfaction about documentation

4.8 Technical Support

Technical support is here defined as a vendor company assisting the customers in solving specific problems. Technical support is normally given by telephone, email or by providing information on the website. While many companies have their own technical support teams [43], most of the CM tools have their own internet community where customers get support from each other in addition to the compant [15].
The overall satisfaction with the technical support was low among the users and the highest level of satisfaction was among the LCFG users (29%) (Fig.4.8). Only 19% of CFEngine users are satisfied with the provided technical support.

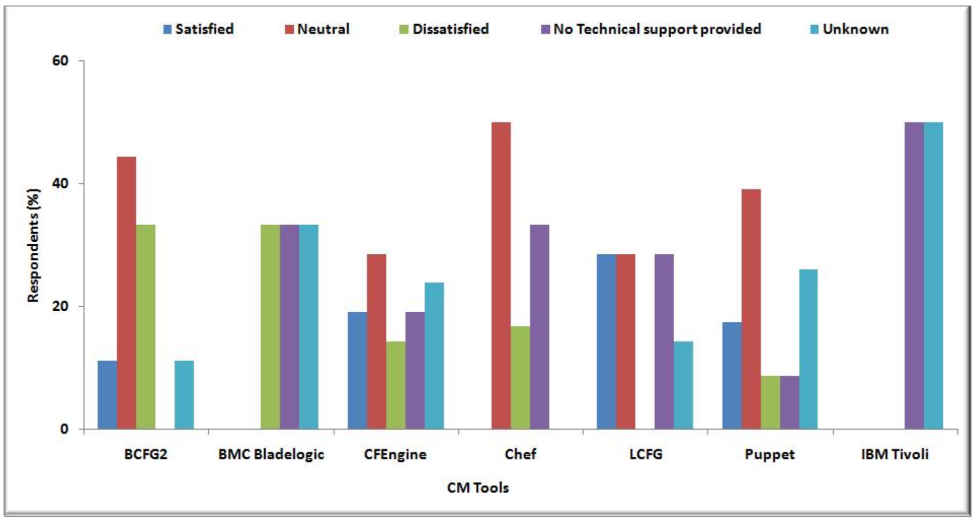


Figure 4.8: Level of satisfaction about technical support

There was also some expressed dissatisfaction among the BCFG2 (33%), BMC Bladelogic (33%), Chef (17%), CFEngine (14%) and Puppet (9%) users (Fig.4.8). This may be that they did not find the provided technical support helpful. It could also be due to lack of communication between the system administrators and the CM tool vendor so the system administrator was not awars about the technical support offered by the vendors of CFEngine, Puppet and Chef [46][47][48].

4.9 Training

Training refers to the CM tool provider or vendor giving training to the customers to improve their ability, potential, competence or performance in operating the CM tool [44].

The overall user satisfaction about trainings was low and the highest satisfaction level was among the Bcfg2 (22%), Chef (17%), LCFG (14%), Puppet (13%) and the CFEngine users (10%) (Fig.4.9). Although CFEngine offers online and classroom training for their customers [47], 10% of the CFEngine users were not satisfied with training provded and 15% claimed that no training was provided at all. Only 13% of the Puppet users were satisfied with the training although Also Puppet often offers three day training programs for the users[60. Thus the reason for this low satisfaction may be due to lack of communication between the system administrators and the CM tool vendors that the vendors of CFEngine, Puppet and Chef offer trainings on these tools [46][47][48]. The highest level of dissatisfaction about trainings was among the Bcgf2 users (22%), but according to the Bcfg2 webpage, no training opportunities are mentiones [45].

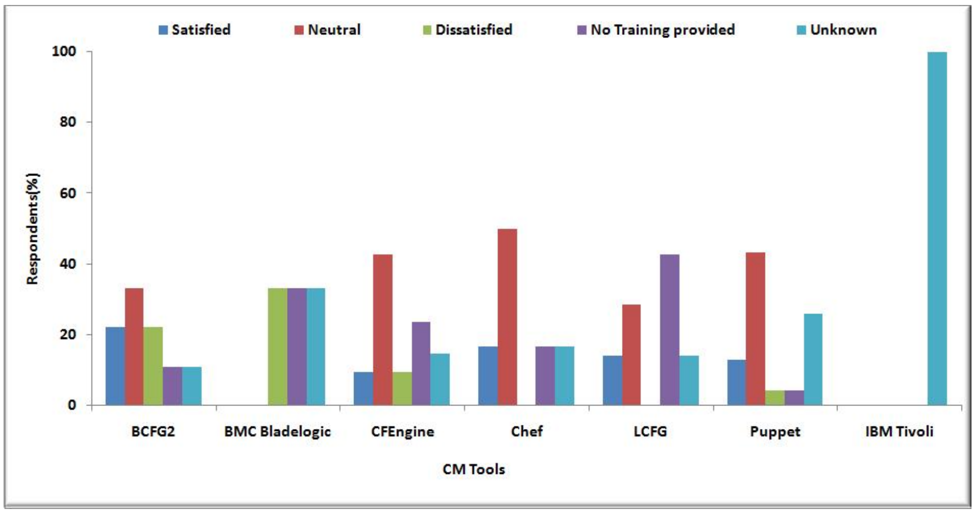


Figure 4.9: Level of satisfaction about training

4.10 Platform Support

Platform refers to core computing System, such as Linux, Solaris, Mac OS X and Windows, on which different types of programs can run.. Traditionally, most of the application programs were developed to run on a specific platform, but current platforms may run multiple application programs for different systems. It is important that the software supports both the CM tool and the platform used [7].

Nearly 100% of the BCM Bladoloic and LCFG users and above 80% of the Bcfg2 Puppet and CFEngine users run their CM tools on LINUX platform(Fig.4.10). Above 60% of all users, except those who used CMC Bandologic, run their CM tool on a Solaris platformThe CM tools used by the respondents are open source based UNIX administrative tools, thus LINUX and Solaris, which are are UNIX based operating systems, makes the automatization of the system easy. MAC OS X is also a popular platform; used by the Bcfg2 (56%), CFEngine (19%), Puppet (43%), Chef (28%) and the LCFG users (43%). A windows based platform were mostly used by the BMC Bladelogic tool users (67%) followed by the users of CFEngine (24%), Puppet (39%), Chef (28%) and Bcfg2 (11%), while none of the respondents using IBM Tivoli and BMC Bladelogic used windows. Most of the tools are running on UNIX based platforms the reason may be like operating system most of the tools are open source software and in most cases it is free to use [54].

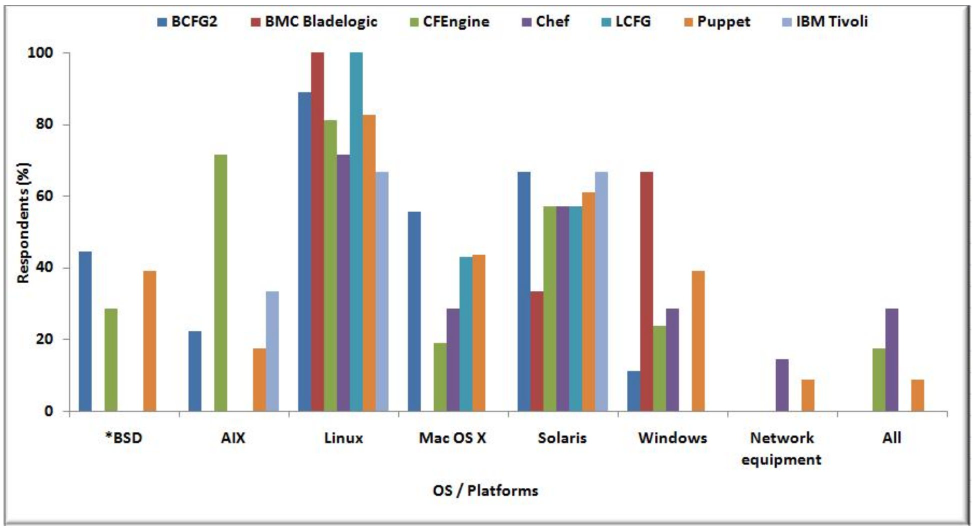


Figure 4.10: Platforms used by the participants

4.11 Access Control

In large IT systems many people may work in the system administration, thus it is important that the main responsible system administrator can regulate these people's access change specific configurations [7].

All CM tools in this research provide access control [7], but only the proportion of Puppet (74%), Chef (67%), CFEngine (57%), LCFG (43%) and BCFG2 (33%) users said they knew that the CM tool they operated had access control (Fig.4.11).

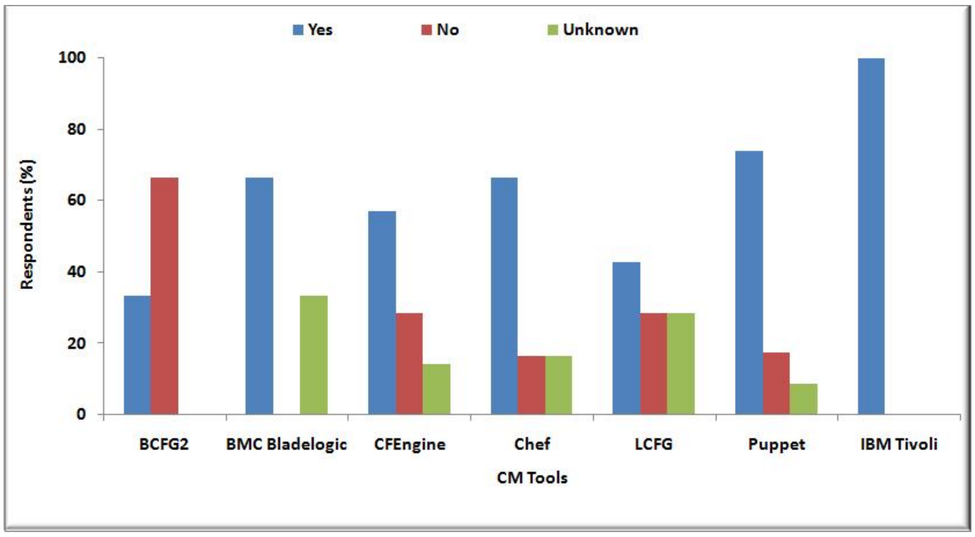


Figure 4.11: Access control facility provided by the CM tools

The reason why not more users were aware of the ability to use access control for their CM tools may be that they were not familiar enough with the term access control or their CM tools, the or that the provided support documents did not inform about this ability.

4.12 Advantages

There are many CM tools available, so to better understand which factors the users base their selection upon, this section will find whether the repondents found the selected factors (lower cost, easy to deploy, documentation, support, flexibility, security, monitoring and inventory tracking) to be an advantage for their CM tool.

The large majority of the CFEngine, Puppet, BCFG2, LCFG and Chef users (81%, 82%, 89%, 71% and 66% of the respectively)it was all, except the BMC Bladelogic, IBM Tivoli and Chef users, found the low cost of their CM tool to be an advantage (Fig.4.12). The deployment procedure was also an important advantage according to the users of Bcfg2, CFEngine Puppet and Chef tools (78%, 43%, 70% and 67% of the respectively).This is because an easy deployment procedure will make it easy to install and configure CM tool, which will save them time. Also flexibility of the CM tool was considered an advantage by All the Chef users and 66%, 77%, 71% and 65% of the CFEngine, BCFG2, LCFG and Puppet users respectively. This is because a flexible tool may make it easier to configure and change the specification according to client and system demands.

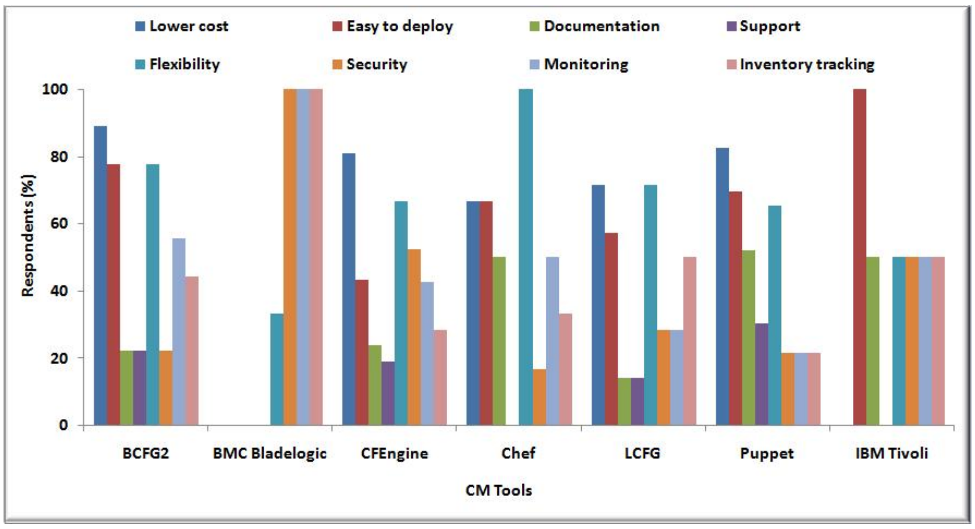


Figure 4.12: Advantages of different CM tools

4.13 Disadvantages

When a decision maker group wants to select a CM tool finally they generally consider not only the advantages of a tool, they also take in mind the disadvantages of the tool. Using this survey the CM tool users asked about the disadvantages of their tools. Not all but a good number of users marked some disadvantages about their tools. The goal of this question was to find the drawback of their tools so that the tool developers can minimize it. Here I have mentioned some common characteristics, which are sometimes may be considered as disadvantages for a tool form user's point of view. Though, it can be vary from user to user.

According to the figure (Fig4.13),showed that only BMC Bladelogic and IBM Tivoli users said that their tool is costly. This may be they need to pay a lot for using this tool. About 44% of BCFG2 users marked that they are missing some functions in their tool. In case of Puppet not a single user said anything about lack of functionality but for CFEngine, LCFG and Chef it was 33%, 28% and 33% respectively. About 83% of Chef user mentioned that they are facing difficulty for the lack of support. In case of, Puppet, BCFG2 and LCFG it was 47%, 66% and 71% respectivelyOn the other hand only 38% of CFEngine users are facing this problem. Good documentation is a very important use for choosing a tool. Like support, the LCFG users are again facing the documentation problem. About 85% of LCFG users a facing this problem but on the other hand only 38% of CFEngine users and 17% of Puppet are facing this disadvantages. All the users of CFEngine considered that CFEngine is fully secure but only 11% of BCFG2 users and only 9% of Puppet user's interest. Most of the people want to mange and configure their tool without giving too much effort, so if a tool is needed too much time to configure and manage, it will be considered as a drawback of that tool. About 71% of CFEngine users said that they need to give too much effort to configure CFEngine. On the other hand, only 17% of Puppet users marked it as a disadvantage for them.

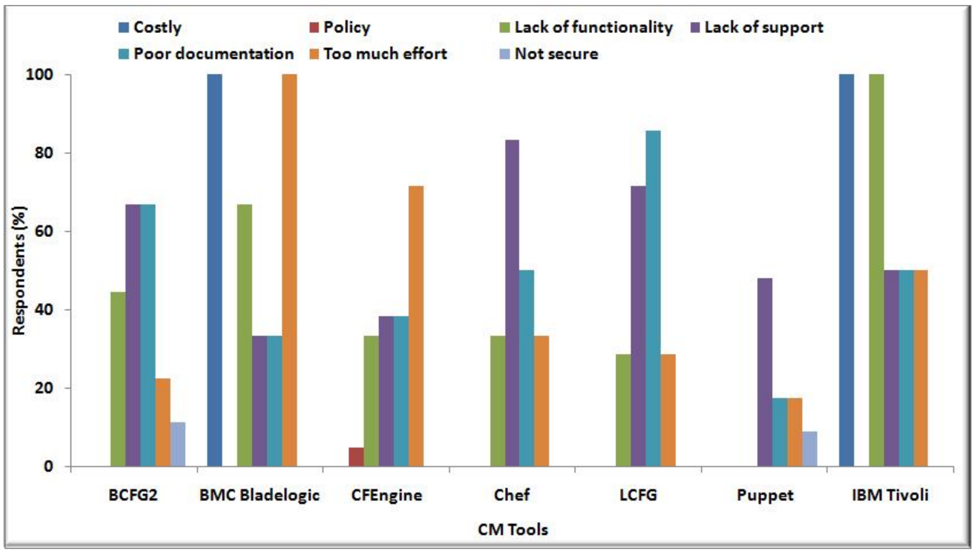


Figure 4.13: Disadvantages of different CM tools

4.14 Satisfaction

The goal of this question was to measure the overall satisfaction level of the CM tool users. If the users are satisfied with their current tool they may not think to switch to another.
Only 14% of the LCFG and 33% of BMCBladelogic users were very dissatisfied with their tool (Fig. 4.14). They may be disatified with the support or find the CM tool difficult to configure or manage. No user of the CFEngine, Puppet, BCFG2 or Chef

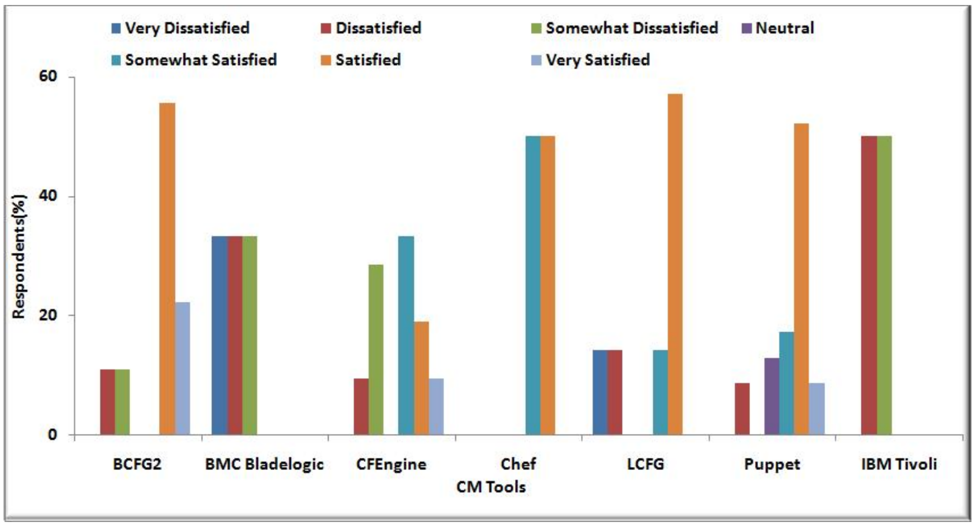


Figure 4.14: Satisfaction level of the respondents about their CM tools

were very dissatisfied with their CM tool and only 9%, 8%, 14% and 11% of the CFEngine, Puppet, LCFG and BCFG2 respectively were dissatisfied with their CM tool. More than 50% of the CFEngine users and 77% of the Puppet users were satisfied (sum of very satisfied, satisfied and somewhat satisfied with their CM tool).

4.15 Deployment Properties

The deployment properties make a configuration management tool or software system usable. The deployment properties was divided into the sub-categories portability, installability, adaptability, configurability, distributability, scalability and stability.

Configurability was considered by the users to be the most important property for CM tool deployment (Fig. 4.15). About 54% of the total respondents found it to be an extremely valuable property for a CM tool and about 35% as very important. This is because it is the collection of most of the functional units and describes the main characteristics of a software or CM tool. Moreover, it influences the system performance and activity [49]. By using the configurability function the users mainly configure their IT system according to the users and organization requirements and policy. Only 2% of the users said that configurability was not valuable for them.

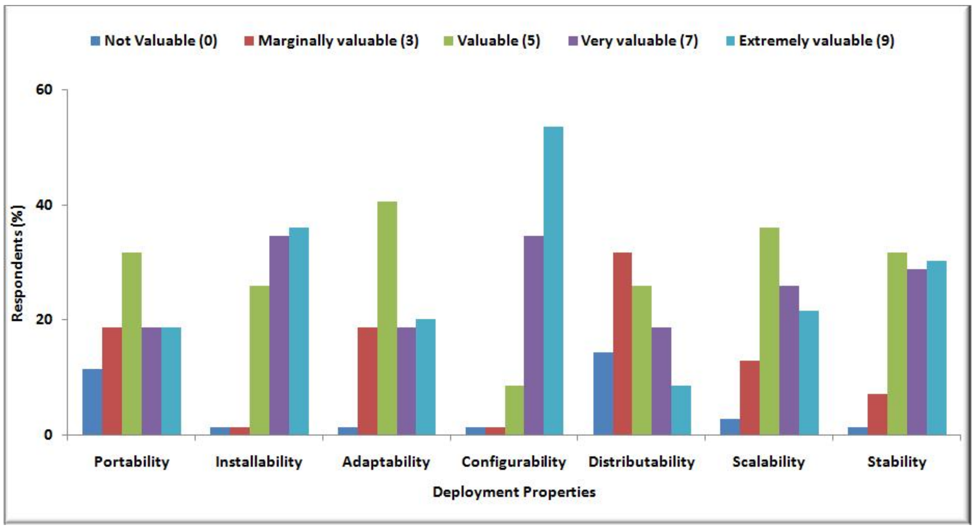


Figure 4.15: Importance of deployment properties

The second most important property for deployment was installability. About 36%, 35% and 26% of the total respondents said installability was extremely valuable, very valuable or valuable for them respectively. The configuration ability depends on the success of the installation. Moreover, many factors are strongly related to the installation process such as platforms, high number of installation steps, total time to install, resource consumption, tools and procedures needed by the installer, ability to uninstall and the present state of running hardware and software.

The third most important property for deployment was stability. About 30%, 29% and 31% of the total participants rated it as extremely valuable, very valuable or valuable respectively (Fig. 4.15). This satisfaction may be due to that if the tool is not stable enough work done will be lost. However, it is difficult to evaluate the stability of a system or tool because it is not only influenced by bugs of the software, but also by factors such as physical memory, disk space or any kind of hardware failure. This may be why only 1% of the users rated stability as not valuable.

The value of portability, adaptability and scalability were rated as almost equal, as 19%, 20% and 21% of the respondents respectively rated these as extremely important (Fig. 4.15). On other hand, 26% of user said that scalability was extremely important, but only 19% found portability and adaptability to be extremely important.

4.16 Specification Management Properties

The specification management properties were divided into the sub categories language, testing, monitoring, versioning and access control so the CM tool developer can focus on characteristics valued by the users.

The most valued specification management property was language, which was considered extremely valuable and very valuable by 23% and 30% of the total respondents respectively (Fig. 4.16). The reason may be, that the users believe that the language of a CM tool should be flexible to substitute their running tools and easy to use so the novice system administrators will be able to operate it [7].About 22% of the total users said language was valuable and marginally valuable and only 3% said that it is not valuable for them. The second most important property for specification management was monitoring. About 10%, 19% and 43% of the total respondents rated it as extremely valuable, very valuable or valuable respectively. The reason may be that a CM tool facilitates the integration with other systems to check or monitor the status of a system or infrastructure.

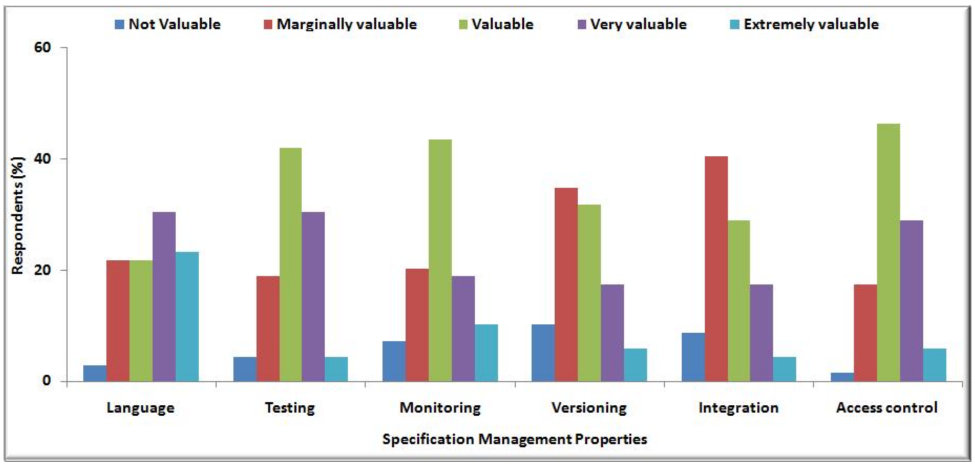


Figure 4.16: Importance of specification management properties

The third most important property was versioning support, found as extremely valuable by only 6% of the total respondents. However, 17%, 32% 35% of the respondents found versioning support to be very valuable, valuable, or marginally valuable respectively (Fig. 4.16).This is because the versioning support helps to save the changes, shows the way to go through the history and helps to roll back to the previous configuration if needed [7]. Access control is also an important property for the user as it helps to protect the system or configuration file to edit or reconfiguration from different levels of users. Thus 29% 46% rated it as very valuable or valuable respectively.

4.17 Support

Support was divided into the subcategories documentation, training and technical support to identify which was the most important from the user's point of view.

From Participant's point of view documentation is the most important property for support (Fig. 4.17). About 45% and 44% of the total respondents found documentation to be extremely valuable and very valuable relatively and only 2% did not find it valuable. This may be because if the provided document is informative and structured they can solve the problems themselves. . The second most import property was technical support, which was extremely valuable, very valuable and valuable for 6%, 20% and 43% of the total users respectively. This may be because when the problem is beyond their control, they seek technical support. Only 5% of the respondents did not think technical support was valuable. The last property, training, was seen as valuable and marginally valuable for 39% and 25% of the total respondents respectively and only 8% said it was not valuable.

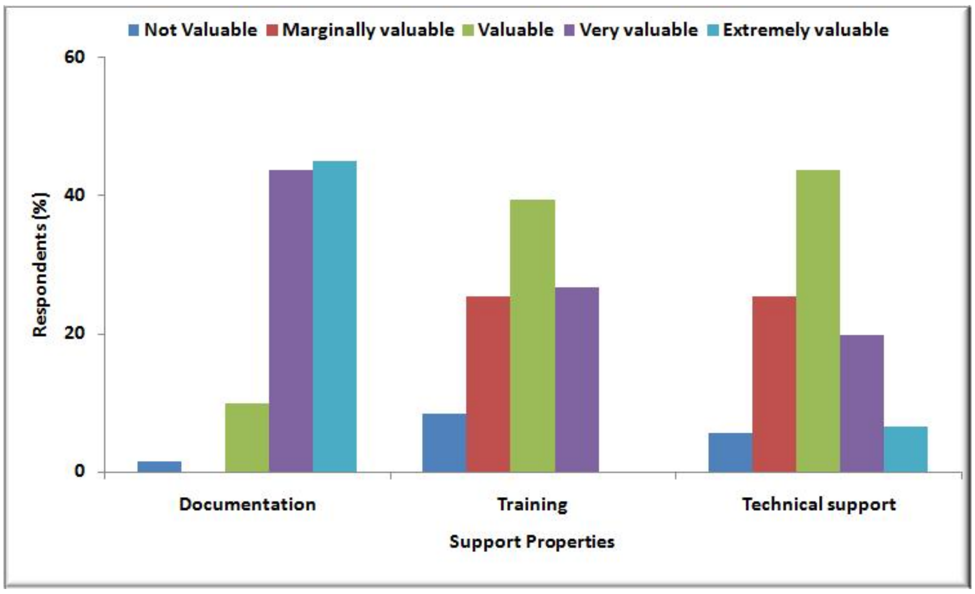


Figure 4.17: Importance of support properties

4.18 Overall result

The most widely used CM tools, used by more than half of the respondents, were CFEngine and Puppet. Chef was the easiest, CFEngine and Puppet moderately and IMB Tivoli the most difficult tool to install. The data showed that installability and configurability were related to each other, thus an easy installation procedure corresponds to an easy configuration procedure. Thus the Chef was the easiest and IBM Tivoli a very difficult CM tool to configure to configure. CFEngine and Puppet were also difficult to configure. CFEngine and Puppet were most scalable, and among them CFEngine was most scalable. IBM Tivoli was the least scalable and LCFG, Chef and BMC Bladelogic were all average regarding scalability. In case of stability, Chef and BMC Bladelogic were the mostandIBM Tivoli the least stable tool. Puppet and CFEngine are also stable. For usability, Chef was the easiest and BMC Bladelogic the most difficult tool use. Most of the users of Chef, LCFG and Puppet were satisfied with their provided document, while a large proportion of the BCFG2 and CFEngine respondents were dissatisfied . However, in the case of technical support CFEngine, Puppet and BMC Bladelogic users were the most satisfied and a large proportion of the BCFG2 and BMC Bladelogic users were dissatisfied. A part of the the CFEngine, Puppet and BCFG2 users were satisfied with the training, whil but at the same time mainly the BCFG2 and BMC Bladelogic users were dissatisfied. Most of the CM tools run on Linux and Solaris platforms, but many respondents also used Mac OS and Windows. All the CM tools provide access control, but still, quite a large proportion of the respondents were not aware about it. Among several advantages, almost all of the respondents appreciated low cost of their CM tool, easiness to deploy and flexibility the most. Next the respondents considered inventory tracking and monitoring the systems as advantages of their CM tools. Among several disadvantages, poor documentation and large efforts needed for deployment were the largest disadvantages, especially among the LCFG, BCFG2 and Chef users. CFEngine, BMC Bladelogic and IBM Tivoli users needed to large efforts for deployment. In case of overall satisfaction of the CM tool, a considerable proportion of CFEngine, Puppet and Chef users were satisfied with their tools, while the larger proportion of users who were dissatisfied abou their CM tools was among the MBC Bladelogic and IBM Tivoli users.

4.19 Selecting the best CM tool according to the respondents

In previous sections single attributes were used to compare the CM tools. In this section the CM tools will be compared based on the aforementioned attributes to find the users perception of the best CM tool.
The data showed 4.1 that CFEngine, Puppet and BCFG2 are mostly used by the users. If the collected data for these three is considered it shows that BCFG2 is easy to use compare to CFEngine and Puppet and Puppet is easy to use compare to CFEngine. On the other hand, CFEngine is more scalable in contrast with Puppet and BCFG2, and Puppet is more scalable than BCFG2. Alternatively, is more stable than CFEngine and BCFG2; conversely CFEngine is more stable than BCFG2. So at this situation if someone wants to select a CM tool then it will difficult for him which one will be the best tool though every tool is cheap. The Analytical Hierarchy Process (AHP) can

help to select the right one.
The table 4.1 below shows that CFEngine acquired the highest score. So CFEngine is the best tool.

Criteria	Support	Deployment	Management	Score
Options	0.077	0.692	0.231	1.000
Puppet	0.604	0.081	0.694	0.263
CFEngine	0.326	0.639	0.253	0.526
BCFG2	0.070	0.279	0.053	0.211
	1.000	1.000	1.000	1.000

Table 4.1: Calculated data for management criteria

Chapter 5

Conclusions

There a many companies using different types of Configuration Management (CM) tools and this research found that CFEngine and Puppet were the most widely used Configuration Management (CM) tools among system administrators. Users select their management tools based on multiple criteria and the overall most desired and important deployment properties were installability and configurability as these make the CM tool easy to use. Scalability and stability were also considered as important attributes. CFEngine is the most scalable and Puppet is the most stable CM tool, which is the reason why these tools are the most commonly used to manage large IT infrastructures.

The most desired specification management properties were language, access control and monitoring and testing properties. The users also consider it to be important that the CM tool vendors provide good customer support and the good customers support provided by the vendors of the CFEngine and Puppet tools is one reason for the widespread use of these tools. The Analytical Hierarchy Process (AHP) algorithm shows that CFEngine was the best Configuration Management tool based on selected criteria.

As the need for CM tools is increasing, so is the need to conduct larger scale research on customer satisfaction of CM tools to provide better information of the different CM tools and make the process to select the CM tool suitable for your need easier and also for the CM tool developers to get feedback from the customer on challenges and potentials for improvement of their products that will ultimately benefit the users.

Bibliography

[1] Burgess, M. Alva Couch, *Modeling Next Generation Configuration Management Tools*. in 20th Large Installation System Administration Conference. 2006. Washington, D.C.

[2] Lueninghoener, C., *Getting Started with Configuration Management*. ;login:, 2011. 36(2): p. 1-6.

[3] Anderson, P., *System Configuration* 2006, Berkeley, USA: USENIX.

[4] Nagaraja, K., et al. *Understanding and Dealing with Operator Mistakes in Internet Services. in Appears in: Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI '04)*. 2004. Piscataway, NJ 08854: Department of Computer Science, Rutgers University.

[5] Patterson, D., D. Oppenheimer, and A. Ganapathi. *Why do Internet Services Fail, and What can be done about it?* in Appears in 4th Usenix Symposium on Internet Technologies and Systems. 2003. Berkeley, USA: University of California at Berkeley, EECS Computer Science Division.

[6] Zheng, W., R. Bianchini, and T.D. Nguyen. *Automatic Configuration of Internet Services*. 2007 [cited 2012 16.05.]; Available from: http://www.cs.rutgers.edu/t̃dnguyen/pubs/eurosys07.pdf

[7] Delaet, T., W. Joosen, and B. Vanbrabant. *A survey of system configuration tools*. [cited 2012 12.04.]; Available from: http://static.usenix.org/event/lisa10/tech/full_papers/Delaet.pdf

[8] PuppetLab. Docs: PE 2.5 *" Overview " About Puppet Enterprise*. 2011 [cited 2012 10.05.]; Available from: http://docs.puppetlabs.com/pe/2.5/overview_about_pe.html

[9] *Configuration Management Tool (GECO)*. 2012 [cited 2012 08.04.]; Available from: http://www.fractal-es.com/Suite/GECO-Eng.pdf

[10] Tsalolikhin, A., *Configuration Management Summit*. ;login:, 2010. 35(5): p. 104-105.

[11] Desai, N., et al., *A Case Study in Configuration Management Tool Deployment*, in 19th Large Installation System Administration Conference 2005, LISA: Mathematics and Computer Science Division, Argonne National Laboratory.

[12] *Comparison of open source configuration management software* [cited 2012 05.04.]; Available from: *http://en.wikipedia.org/wiki/Comparison_of_open_source_configuration_management_software*

[13] *Bcfg2 documentation* 1.2.2. [cited 2012. 01.03]; Available from: http://docs.bcfg2.org/

[14] Anderson, P., *Why Configuration Management Is Crucial ;login:, 2006. 31(1): p. 5-8.*

[15] *CFEngine and Other Configuration Management Software.* 2010 [cited 2012 03.04.]; Available from: https://cfengine.com/manuals_files/SpecialTopic_Comparison.pdf.

[16] Rajneesh, *Cfengine 3 Beginner's Guide*, 2011, Birmingham, UK: Packt.

[17] Dart, S., *Spectrum of Functionality in Configuration Management Systems*, 1990, Software Engineering Institute, Carnegie Mellon University: Pittsburgh, Pennsylvania.

[18] *Modern SaaS IT Service Desk can help reduce support costs*, [cited 2012 01.03.]; Available from: http://www.samanage.com/blog/.

[19] Anderson, P., *System Administration: Large-Scale Linux Configuration Management*. Linux Journal 2000. 19(72).

[20] *Configuration management* [cited 2012 05.05.]; Available from: http://en.wikipedia.org/wiki/Configuration_management#cite_note-0.

[21] *IEEE Guide to Software Configuration Management*, [cited 2012 03.05.]; Available from: http://standards.ieee.org/findstds/standard/1042-1987.html.

[22] Dart, S.A., *The Past, Present, and Future of Configuration Management*, 1992 [cited 2012 07.05.]; Available from: ftp://ftp.sei.cmu.edu/pub/case-env/config_mgt/tech_rep/cm_past_pres_future_TR08_92.pdf.

[23] *Configuration.* [cited 2012 01.04.]; Available from: http://www.webopedia.com/TERM/C/configuration.html.

[24] Dart, S., *Concepts in Configuration Management Systems*, [cited 2012 02.05.]; Available from: http://sceweb.uhcl.edu/boetticher/swen5230/concepts-in-configuration-management.pdf.

[25] Kan, S.H., *Metrics and Models in Software Quality Engineering*, Second Edition 2002: Addison-Wesley Professional.

[26] Jones, C., *Critical Problems In Software Measurement*, 1993: Information Systems Management.

[27] Regedor, M., D. Cruz, and P. Henriques., *The role of best practices in assessing software quality*. [cited 2012 09.04.]; Available from: http://opencert.iist.unu.edu/Papers/2011-paper-S1-B.pdf.

[28] Kan, S.H., V.R. Basili, and L.N. Sapiro, *Software quality: An overview from the perspective of total quality management*. IBM System Journal, 1994. 33(1): p. 4-19.

[29] Crosby, P. B., *"Quality is free : the art of making quality certain"*,McGraw-Hill

[30] Berander, P., et al., *Software quality attributes and trade-offs*, ed. L. Lundberg, M. Mattsson, and C. Wohlin2005, Karlskrona: Blekinge Institute of Technology.

[31] Brown, N., *Little Book of Configuration Management* 1998: AIRLIE.

[32] *Instability guidelines*. [cited 2012 03.04.]; Available from: http://www.testingstandards.co.uk/installability_guidelines.htm.

[33] Turnbull, J. and J. McCune, *Pro Puppet*, 2011, Paul Manning.

[34] *Chef*. [cited 2012 05.05.]; Available from: http://www.opscode.com/chef/.

[35] *Computer configuration* [cited 2012 03.04.]; Available from: http://en.wikipedia.org/wiki/Computer_configuration.

[36] *OIT Campus Linux Services Wiki*. [cited 2012 01.05.]; Available from: https://secure.linux.ncsu.edu/moin/Bcfg2/FAQ

[37] *Bcfg2 Configuration*. [cited 2012 04.05.]; Available from: http://zenit.senecac.on.ca/wiki/index.php/Bcfg2_Configuration#Server_Install

[38] *Scalability* [cited 2012 25.04.]; Available from: http://en.wikipedia.org/wiki/Scalability

[39] Stephen, S., S. Yau, and J.S. Collofello, *Some Stability Measures for Software Maintenance LEEE Transactions On Software Engineering*, 1980. 6(6): p. 545-552.

[40] Chiang, C., *Software Stability in Software Reengineering in Information Reuse and Integration*, 2007. IRI 2007. IEEE International Conference on2007, IEEE Xplore: Arkansas University.

[41] *Usability* [cited 2012 05.05.]; Available from: http://en.wikipedia.org/wiki/Usability

[42] *Documentation Definition*. [cited 2012 03.04.]; Available from: http://www.linfo.org/documentation.html

[43] *Technical support*. [cited 2012 06.05.]; Available from: http://en.wikipedia.org/wiki/Technical_support

[44] *Training* [cited 2012 01.03.]; Available from: http://en.wikipedia.org/wiki/Training

[45] *Bcfg2*, [cited 2012 01.03.]; http://trac.mcs.anl.gov/projects/bcfg2

[46] *Training*. [cited 2012 01.03.]; Available from: http://www.opscode.com/training/

[47] *CFEngine Support*. [cited 2012 01.03.]; Available from: http://www.cfengine.com/support

[48] http://puppetlabs.com/category/events/ , visisted on 21, 2012

[49] *Computer configuration* [cited 2012 05.04.]; Available from: http://en.wikipedia.org/wiki/Computer_configuration

[50] Al-qutaish, R., M. Muhairat, and B. Al-kasasbeh, *The analytical hierarchy process as a tool to select open source software*, in Proceedings of the 8th WSEAS Int. Conference on SOFTWARE ENGINEERING, PARALLEL and DISTRIBUTED SYSTEMS, Al-Zaytoonah University of Jordan: Jordan.

[51] Sanga, C. and I.M. Venter, *Is a Multi-Criteria Evaluation Tool Reserved for Experts?*, The Electronic Journal Information Systems Evaluation, 2009. 12(1): p. 165 -176.

[52] Render, B., R.M. Stair, and M.E. Hanna, *Quantitative Analysis for Management*, ed. 7th2000, Newyork: Person.

[53] Kunz, J. *The Analytic Hierarchy Process (AHP)*, 2010 [cited 2012 05.04.]; Available from: http://www.cityofeagle.com/vertical/Sites/%7B78557FDD-14BE-414E-8624-C15ED40E9C6A%7D/uploads/%7B85436A40-9869-4F45-B0CA-6B2539A26C81%7D.PDF

[54] *Open-source software*, [cited 2012 05.04.] http://en.wikipedia.org/wiki/Open-source_software

Appendix

APPENDIX: 1
The AHP method consists of three different levels. The top level (here select the best CM tool) gives the decision of which on is the best. The middle level illustrates all the criteria or selected criteria (here support, deployment, management) which are considered for this purpose and the lower level describes the alternative options (here Puppet, CFEngine and BCFG2)

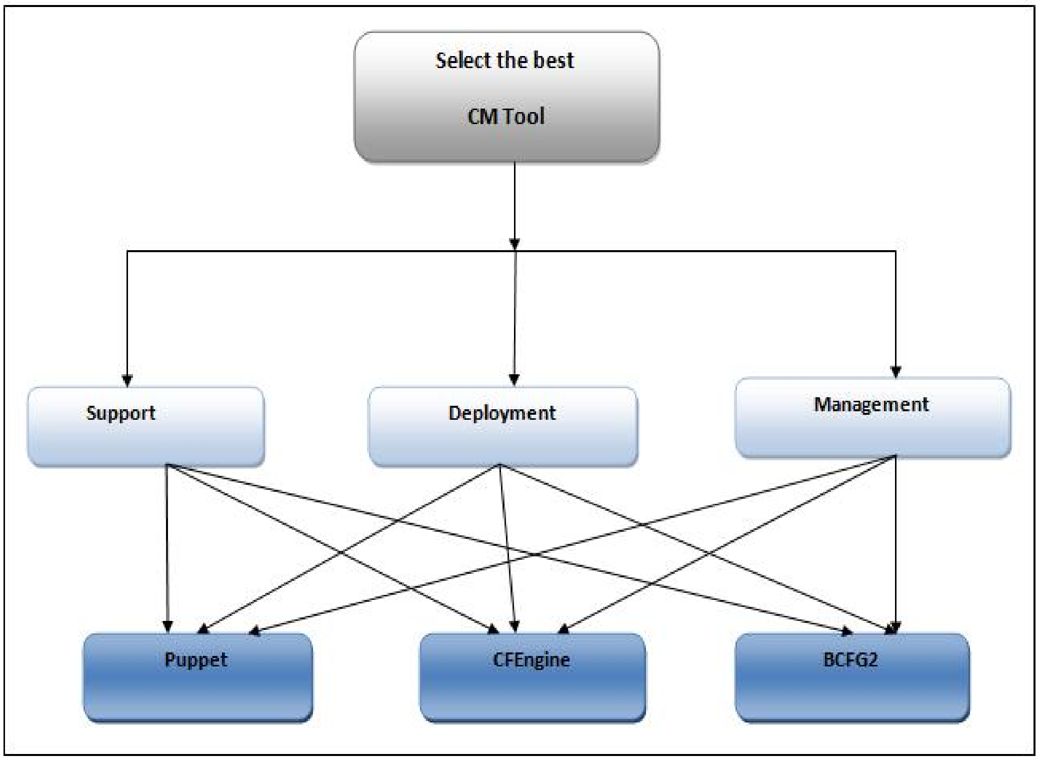


Figure 5.1: AHP method decision hierarchy

A survey was conducted on CM tool and according to the user's opinion and in contrasting the three criteria - support, deployment and management; deployment property is the most important criteria for a CM tool. Deployment is "Absolutely more important (9)" than support and Deployment is "Somewhat more important (3)" than Management .Comparing management with support; determined that management is more important. Specially, Management is "Somewhat more important(3)" than support. The given table below shows the corresponding values.

Intensity of importance	Definition
1	Equally importance
3	Somewhat more importance
5	Much more importance
7	Very much more importance
9	Absolutely more important
2,4,6,8	Intermediate values

Table 5.1: Scale of relative importance of factors

Equation (I)

$$\begin{array}{cccc} & Criterion1 & Criterion2 & Criterion3 \\ Criterion1 & \frac{c1}{c1}=a_{11} & \frac{c1}{c2}=a_{12} & \frac{c1}{c3}=a_{13} \\ Criterion2 & \frac{c2}{c1}=a_{21} & \frac{c2}{c2}=a_{22} & \frac{c2}{c3}=a_{23} \\ Criterion3 & \frac{c3}{c1}=a_{31} & \frac{c3}{c2}=a_{32} & \frac{c3}{c3}=a_{13} \end{array}$$

With the abovementioned pairwise comparison values a matrix is constructed using the equation (1). In the matrix the principal diagonal have values of 1, because every factor is as valuable as itself.

	Support	Deployment	Management
Support	1.000	0.111	0.333
Deployment	9.000	1.000	3.000
Management	3.000	0.333	1.000

Now the nth root of the three criteria is calculated.
Support: (1.000 * 0.111 * 0.333) = (0.037)$^{(1/3)} = 0.333$
Deployment: (9.000 * 1.000 * 3.000) = (27)$^{(1/3)} = 3.000$
Management : (3.000*0.333*1.000)= (1)$^{(1/3)} = 1$
After calculating the 3rd root of every criteria, the calculated values was added.

	Support	Deployment	Management	$3^{rd} root of criteria$
Support	1.000	0.111	0.333	0.333
Deployment	9.000	1.000	3.000	3.000
Management	3.000	0.333	1.000	1.000
				4.333

After calculating the pairwise matrix, the priority vector was calculated and it was calculated by normalize the matrices. The equation is as below:
Equation (II)
By using the abovementioned equation priority vector was calculated:
Support: (0.333/4.333) = 0.077
Deployment: (3.000/4.333) = 0.692
Management: (1/4.333) = 0.231

$$a'_{ij} = a_{ij} \Big/ \textstyle\sum_{i=1}^{n} a_{ij} \quad , i,j=1,2,\ldots,\text{n}$$

$$\begin{array}{llll}
\textit{Objective} & \textit{criterion } 1 & \textit{criterion } 2 & \textit{criterion } 3 \\
\textit{criterion } 1 & \dfrac{a_{11}}{\sum_{i=1,j=1}^{3} a_{ij}} & \dfrac{a_{12}}{\sum_{i=1,j=2}^{3} a_{ij}} & \dfrac{a_{13}}{\sum_{i=1,j=3}^{3} a_{ij}} \\
\textit{criterion } 2 & \dfrac{a_{21}}{\sum_{i=1,j=1}^{3} a_{ij}} & \dfrac{a_{22}}{\sum_{i=1,j=2}^{3} a_{ij}} & \dfrac{a_{23}}{\sum_{i=1,j=3}^{3} a_{ij}} \\
\textit{criterion } 3 & \dfrac{a_{31}}{\sum_{i=1,j=1}^{3} a_{ij}} & \dfrac{a_{32}}{\sum_{i=1,j=2}^{3} a_{ij}} & \dfrac{a_{33}}{\sum_{i=1,j=3}^{3} a_{ij}}
\end{array}$$

	Support	Deployment	Management	$3^{rd}root$ of criteria	Priority Vector(VC)
Support	1.000	0.111	0.333	0.333	0.077
Deployment	9.000	1.000	3.000	3.000	0.692
Management	3.000	0.333	1.000	1.000	0.231
				4.333	1.000

At this stage the Consistency Ratio (CR) was calculated. Calculating Consistency Ratio is a four step procedure:

Procedure one: The values of each column are added and then the calculated sum is multiplied by the respective weight. It is computed using the following formula:

AW^{T}..(III)
Where A is the pairwise comparison matrix and W$^{T} istheweightvector$.

Support : (1.000 + 9.000 + 3.000) = 13.000 * 0.077 = 1.001

Deployment : (0.111 + 1.000 + 0.333) = 1.444 * 0.692 = 0.999

Management = (0.333 + 3.000 + 1.000) = 0.999 * 0.231 = 0.231

	Support	Deployment	Management	$3^{rd} root$ of criteria	Priority Vector(VC)
Support	1.000	0.111	0.333	0.333	0.077
Deployment	9.000	1.000	3.000	3.000	0.692
Management	3.000	0.333	1.000	1.000	0.231
Sum	13.000	1.444	4.333	4.333	1.000
Sum*PV	1.001	0.999	1.001	3.001	

Procedure two: The values in the Sum*Pv row are added together (1.001 + 0.999 + 0.231) = 3.001 and put it under the 3rd root of criteria column. This is the value of lambda-max.

$$\lambda_{\max} = \frac{1}{n}\sum_{i=1}^{n}\frac{(Aw^T)_i}{(w^T)_i}$$

Procedure three: At this stage the Consistency Index (CI) is calculated using the formula:

$$CI = \frac{\lambda_{\max} - n}{n - 1}$$

Where n is the number of criteria. Here n=3.
CI= (3.001 - 3)/ (3 - 1) = 0.001/2 = 0.001
Procedure Four: Now the Consistency Ratio (CR) is calculated and it is computed by dividing the Consistency Index (CI) by Random Index (RI).

Consistency Ratio (CR) = Consistency Index (CI) / Random Index (RI)

= 0.001/0.58

= 0.001

n	1	2	3	4	5	6	7	8	9	10
RI	0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49

Table 5.2: Random Index

	Support	Deployment	Management	$3^{rd} root$ of criteria	Priority Vector(VC)
Support	1.000	0.111	0.333	0.333	0.077
Deployment	9.000	1.000	3.000	3.000	0.692
Management	3.000	0.333	1.000	1.000	0.231
Sum	13.000	1.444	4.333	4.333	1.000
Sum*PV	1.001	0.999	1.001	3.001	
Lambda max	3.001				
CI	0.001				
CR	0.001				

Table 5.3: Calculated data for Support criteria

If Consistency Index (CI) = 0 that means the pairwise comparison matrix is seriously consistence. The Consistency Ratio illustrates how much consistence the decision maker; while taking the pairwise comparison. If CR <= 0.10, the pairwise comparison matrix is consistence enough and if CR >= 0.10, the pairwise comparison matrix is seriously consistence.

At this stage, need to develop the rating for every selection option for every criterion For this purpose, regarding support, I determined that the support for CM tool Puppet is "Equally to somewhat more important (2)" to CM tool CFEngine, support for CM tool Puppet is "Very much to absolutely more important (8)" to CM tool BCFG2 and support for CM tool CFEngine is "Much more important (5)" to CM tool BCFG2.

	Puppet	CFEngine	BCFG2	$3^{rd} root$ of criteria	Priority Vector(VC)
Puppet	1.000	2.000	8.000	2.519	0.604
CFEngine	0.500	1.000	5.000	1.357	0.326
BCFG2	0.125	0.200	1.000	0.292	0.070
Sum	1.625	3.200	14.000	4.168	1.000
Sum*PV	0.982	1.043	0.980	3.005	
Lambda max	3.005				
CI	0.003				
CR	0.005				

Table 5.4: Calculated data for Support criteria

Regarding deployment, determined that the deployment for CM tool CFEngine is "Somewhat more important (3)" to CM tool Puppet, deployment for CM tool BCFG2 is "Absolutely more important (9)" to CM tool Puppet and deployment for CM tool CFEngine is "Much more to very much more important (6)" to CM tool BCFG2.

	Puppet	CFEngine	BCFG2	$3^{rd} root$ of criteria	Priority Vector(VC)
Puppet	1.000	0.333	0.111	0.333	0.081
CFEngine	3.000	1.000	6.000	2.621	0.639
BCFG2	9.000	0.167	1.000	1.145	0.279
Sum	13.000	1.500	7.111	4.099	1.000
Sum*PV	1.053	0.959	1.984	3.996	
Lambda max	3.996				
CI	0.498				
CR	0.859				

Table 5.5: Calculated data for deployment criteria

Determined that management for CM tool Puppet is "Somewhat to much more important (4) "to CFEngine, CM tool Puppet is "Absolutely more important (9)" to BCFG2 and CM tool CFEngine "Very much more important (7)" to CM tool BCFG2.

	Puppet	CFEngine	BCFG2	$3^{rd} root$ of criteria	Priority Vector(VC)
Puppet	1.000	4.000	9.000	3.302	0.694
CFEngine	0.250	1.000	7.000	1.205	0.253
BCFG2	0.111	0.143	1.000	0.252	0.053
Sum	1.361	5.143	17.000	4.759	1.000
Sum*PV	0.945	1.301	1.901	3.147	
Lambda max	3.147				
CI	0.074				
CR	0.128				

Table 5.6: Calculated data for management criteria

Now to select the best CM tool need to compute the weighted average rating.
To find out the score for each CM tool, score was calculated by the following way.
Puppet = (0.077 * 0.604) + (0.692 * 0.081) + (0.231 * 0.694) = 0.263
CFEngine =(0.077 * 0.326) + (0.692 * 0.639) + (0.231 * 253) = 0.526
BCFG2 = (0.077 * 0.070) + (0.692 * 0.279) + (0.231 * 0.053) = 0.211

Criteria	**Support**	**Deployment**	**Management**	**Score**
Options	0.077	0.692	0.231	1.000
Puppet	0.604	0.081	0.694	0.263
CFEngine	0.326	0.639	0.253	0.526
BCFG2	0.070	0.279	0.053	0.211
	1.000	1.000	1.000	1.000

Table 5.7: Calculated data for management criteria

The table shows that CFEngine acquired the highest score. So CFEngine is the best tool.

APPENDIX: 2
A SURVEY ON CONFIGURATION MANAGEMENT TOOL

1: Which Configuration Management tool do you use? (Please tick in the circle):
◯$CFEngine$
◯ $Puppet$
◯ $Chef$
◯ $Bcfg2$
◯ $Lcfg$
◯ $IBMTivoli$
◯ $MicrosoftSCM$

If Others, Please write down the name:...

2: Installation, Please give the following levels (tick in the circle):
◯$Verydifficult$
◯ $Difficult$
◯ $Slightlydifficult$
◯ $Easy$
◯ $Veryeasy$

3: Configuration, Please give the following levels (tick in the circle):
◯$Verydifficult$
◯ $Difficult$
◯ $Slightlydifficult$
◯ $Easy$
◯ $Veryeasy$

4: Scalability (tick in the circle):
◯ < 100
◯ $100 - 1000$
◯ $1000 - 10000$
◯ $10000 - 100000$
◯ > 100000

5. Stability (tick in the circle):
◯*Verystable*
◯ *Stable*
◯ *Unstable*
◯ *Veryunstable*
◯ *Unknown*

6: Usability (tick in the circle):
◯*Verydifficult*
◯ *Difficult*
◯ *Slightlydifficult*
◯ *Easy*
◯ *Veryeasy*

7: Documentation (tick in the circle):
◯*Satisfied*
◯ *Neutral*
◯ *Dissatisfied*
◯ *Nodocumentationprovided*
◯ *Unknown*

8: Technical Support (tick in the circle):
◯*Satisfied*
◯ *Neutral*
◯ *Dissatisfied*
◯ *NoTechnicalSupportprovided*
◯ *Unknown*

9: Training (tick in the circle):

◯ $Satisfied$
◯ $Neutral$
◯ $Dissatisfied$
◯ $NoTrainingprovided$
◯ $Unknown$

10: Platform (One/multiple answer):

◯ $*BSD$
◯ AIX
◯ $LINUX$
◯ $MacOsX$
◯ $Solaris$
◯ $Windows$
◯ All

11: Provide access control facility?

◯ Yes
◯ No
◯ $Unknown$

12: Advantages of your CM tool (please tick in the circle):

◯ $Lowercost$
◯ $Easytodeploy$
◯ $Documentation$
◯ $Support$
◯ $Flexibility$
◯ $Security$
◯ $Monitoring$
◯ $Inventory$
◯ $Tracking$

13: Disadvantages of your CM tool (please tick in the circle):

◯$Costly$
◯ $Policy$
◯ $Lackoffunctionality$
◯ $Lackofsupport$
◯ $Poordocumentation$
◯ $Toomucheffort$
◯ $Notsecure$

14: What is your level of satisfaction with your CM tool? (Please tick in the circle):
◯$Verydissatisfied$
◯ $Dissatisfied$
◯ $Somewhatdissatisfied$
◯ $Neutral$
◯ $Somewhatsatisfied$
◯ $Satisfied$
◯ $Verysatisfied$

15: Please Rank according to the importance (Write down the number in bracket): 9=Extremely valuable; **7=Very valuable**; **5= Valuable**; **3= Marginally valuable; 0=Not valuable**
15.1: Deployment Properties:
◯$Portability$ [.....]
◯ $Installability$ [.....]
◯ $Adaptability$ [.....]
◯ $Configurability$ [.....]
◯ $Distributability$ [.....]
◯ $Scalability$ [.....]
◯ $Stability$ [.....]

15: Please Rank according to the importance (Write down the number in bracket):
9=Extremely valuable; 7=Very valuable; 5= Valuable; 3= Marginally valuable; 0=Not valuable
15.2: Specification Management Properties:
◯*Language* [.....]
◯ *Testing* [.....]
◯ *Monitoring* [.....]
◯ *Versioning* [.....]
◯ *Integration* [.....]
◯ *Accesscontrol* [.....]

15: Please Rank according to the importance (Write down the number in bracket):
9=Extremely valuable; 7=Very valuable; 5= Valuable; 3= Marginally valuable; 0=Not valuable

15.3: Support:
◯*Documentation* [.....]
◯ *Training* [.....]
◯ *Technicalsupport* [.....]