Helsinki University of Technology
Mat-2.177: Seminar on case studies in operations research
Spring 2005

Analysis of the Utility of Office Applications

Final Report

24.4.2005

Renjish Kaleelazhicathu 61896R
Tuomas Kuronen 55028E
Ilkka Leppänen 55392J
Ilkka Hirvonen 51555K

List of figures

Abstract

Utility of office applications is becoming increasingly important as the technology of computing devices such as PCs develops and the demand for highly sophisticated and efficient Office applications grows. This study was conducted by a group of students at the Helsinki University of Technology, as a project work for the Seminar on Case Studies in Operations Research. The project work was conducted with the support of Nokia Research Center.

Our main objective in this study was to evaluate the utility of various Office applications with respect to their resource requirements constraint. This study might give some clue whether the utility that the user experiences from the applications (s)he uses has risen in recent years. The model developed and results achieved based on the model should serve as a reference point for understanding the relationship between the mobile Office applications and their resource requirements.

Keywords: Utility, resource requirements, GOMS-KLM, usability, software bloat.

Executive Summary

The report is the result of a case study titled "Analysis of the utility of office applications", spanning three months, conducted by students from Helsinki University of Technology, in association with Nokia Research Centre for the course Mat-2.177: seminar on case studies in operations research.

The major research questions answered in this report are:

- What is an appropriate method to measure end-user's utility from an Office application?
- How has the utility changed in the previous versions of an Office application?
- What is the shape of utility?
- Where does it go in future?
- What is the behaviour of the utility-resource ratio?

In order to answer these questions, we developed a multi-attribute model. The attribute identification and score calculations were based on sources from literature, discussions among our team and supervisors. Many of our assumptions were validated on the basis of our questionnaire results. The limitations of our model are also mentioned in this report.

The customer segment considered in our study is business customer. We chose word processors as an example of the Office application for our analysis. Our model was applied on two different cases.

Case I: Comparison of different versions of a word processor.
In this case, we considered three different versions of MS Word (v2.0, v97 and v2002) over a period of 10 years. The main results from this case show a decrease in the utility over this period while the resource requirements show considerable increase. The phenomenon of software bloat is also witnessed.

Case II: Comparison of different word processor applications
In this case, we considered three different word processors (MS Word 2002, OpenOffice (v1.1.2) and AbiWord). The main results from this case shows that OpenOffice has the highest utility among the three followed by AbiWord and MS Word. In terms of the utility-resource ratio, AbiWord demonstrates the highest value followed by OpenOffice and MS Word.

The report also presents results from our questionnaire survey answered by 60 participants, mostly students from HUT.

1 Introduction

This report presents the case study titled "Analysis of the utility of office applications" conducted as part of the course Mat-2.177 seminar on case studies in operations research in association with Nokia Research Centre.

1.1 Motivation

Technology is increasingly becoming an integral part of our lives today. This has increased the requirements for technology vendors to develop successful products. These requirements are not limited only to the technological innovations. A vendor's ability to provide design solutions that enhance properties such as ease of use and simplicity is key to the success. In other words, a user-centered approach is the primary factor for greater acceptance of any technology of today. This holds true in the case of all software applications as well.

In recent years, office applications in PCs such as word processors, spread sheets and presentations have gained wider acceptance mainly among business consumers and student community due to their ability to provide multiple functionalities and ease of use. However, an increase in functionalities with greater automation has also brought about higher usage of processing resources. In some cases, the additional functionalities are claimed to add minor utility to the end-user vis-à-vis the resources used.

With the recent popularity of nomadic and mobile terminals such as the PDAs and 2G/3G phones respectively, availability of office applications on these terminals are considered as part of the natural progression from the PC environment. However, the processing power of PCs have been growing in accordance with Moore's law[α] [1] over the past years making the resource utilization not a major bottleneck unlike in the case of PDAs and mobile phones where resource optimization in terms of memory and CPU usage is a major priority.

1.2 Objective

The main objective of this study is to evaluate the utility of various Office applications with respect to their respective resource utilizations in a PC environment, conduct analysis and draw meaningful conclusions for the migration of office applications to resource-constrained terminals. Through this study, we aim to provide a generic framework for utility and resource requirement measurements in the case of Office software applications such as word processors.

[α] Moore's law is an empirical observation stating, in effect, that at the current rate of technological development and advances in the semiconductor industry, the complexity of integrated circuits doubles every 18 months. It has a straightforward effect also to the development of processing power of computers.

1.3 Scope

The scope of this study includes the following:

- Development of a framework for utility and resource measurement of office applications.
- Comparison of the utility vs. resource required for office applications identified in this study.
- Comparison of utility vs. resource required for different versions of the same application during a time period (For instance: comparing different versions of Microsoft Word)
- Plotting the utility curve and locate the current utility of applications on this curve.
- Analysis of the results obtained from the utility-resource model vis-à-vis a questionnaire.

Besides this, the report provides references of previous work done in this area that could be of benefit for extending this study and framework in the future.

1.4 Terminology

CPM-GOMS: Cognitive Perceptual Motor GOMS, a cognitive modeling technique based on GOMS that emphasizes parallel activities.

GOMS: Goals, Operators, Methods and Selection rules is a technique to estimate usability.

GUI: Graphic User Interface such as the desktop environments provided for different operating systems.

HCI: Human-Computer Interaction.

IFPUG: International Function Point Users Group

KLM: Keystroke-Level Model is the simple version of GOMS used for calculating usability.

NGOMSL: Natural GOMS language is another version of GOMS. In addition to execution time, this technique also provides learning time predictions.

Office applications: Application suite used for business computing and analysis such as word processors, spreadsheets etc.

PC: Personal Computer

PDA: Personal Digital Assistant

Resource requirements: Computing device's resource requirements for executing a software application.

SMARTER: Simple Multi-Attribute Rating Technique Exploiting Rank is a weighting method that is used in value trees. This method is an enhancement of the original method of Simple Multi-Attribute Rating Technique (SMART). In SMARTER, you rank the attributes in order of importance, and from this order you calculate mean weights. These criteria sum to one.

Utility: Utility has multiple definitions in different contexts. It is originally a term of economics meaning a measure of happiness gained from a good or service. In decision science, utility is a concept that is usually used when making decision under uncertainty; then, utility is a value of making decisions under risk [2]. In this document, we use utility as a measure of the value that the user gains from the usage of some software application compared to another.

WP: Word Processor such as MS Word.

1.5 Organization of the report

The organization of this report is as follows:

Chapter 2 provides an overview of background and related work available in the literature. The literature survey mentioned in this chapter was the basis for our model development.

Chapter 3 describes our model in detail. It explains the methodology adopted in our study, the calculation procedures adopted for normalization and ranking, the different attributes identified for utility and resource measurements followed by a discussion on the limitations of our model.

Chapter 4 provides the results obtained by applying our model and their analysis. We applied our model on two different cases. Another contribution in this chapter is the results and analysis of the questionnaire that we developed for the purpose of validation of some of our assumptions and to achieve additional insight. The chapter also presents our thoughts on calculating the usability scores for a document and the difficulties we encountered in this regard. This is followed by a brief discussion on the time spent on doing various tasks along the course of this study.

Chapter 5 concludes this report by providing a summary of our main contributions, results, suggestions for future work and some caveats. The chapter also includes some thoughts on utility and resource issues in using Office applications in a mobile environment.

The questionnaire and the related comments received from participants are provided as **Appendix** for reference.

2 Background and related work

The utility of a software application largely falls under the domain of HCI. There is a growing number of literatures available in this area. However, much of these are either specific to the application design and development stage or are limited in the scope of calculating the utility from an end-user's perspective. In other words, based on our review, no single framework or model is sufficient enough to evaluate the utility of software applications, especially those that require a higher degree of human-computer interaction such as office application suite. Hence, our approach in this regard has been to identify different attributes that contribute to the end-user utility and combine them to generate a single function. Some of the approaches and related work in the area of HCI that we came across during the literature review phase are presented in this section.

Function Points [3] provide an estimate of the size of a software application. The size is estimated based on the number of functions in the application. Function Points consists mainly of five metrics: 1) Number of inputs, 2) Number of outputs 3) Number of inquiries 4) Number of data files and 5) Number of interfaces. A major benefit of this technique is that it is independent of the language, development methodology or technology used to develop the application. IBM was a pioneer in formulating this technique in the late seventies which has since then been refined. IFPUG [4], founded in the late eighties maintains Counting Practice Manuals for function point analysis. Other benefits of this technique include measuring productivity, estimating development and support, monitoring outsourcing agreements, enabling IT related business decisions and normalizing other measures. Functions Point technique can be used for GUI-based as well as object-oriented software applications. However, we realized that this technique is quite close to software-design stage (includes back-end operations) and doesn't necessarily capture the utility from an end-user's perspective. Moreover, the function point counting requires many hours of practice and specialized consultants are used to conduct such analysis. Considering our period of the project, this wasn't considered feasible.

Feature Points [5] provide an estimate of the size of a software application that has higher algorithmic complexity such as real-time systems, system software, embedded systems, communication systems etc. Feature Point, developed by Software Productivity Research in 1986, is considered as a superset of Function Point metric since it adds an additional metric: Number of algorithms, to include the software complexity. We realized that this technique, like Function Point, is more relevant for system design stage than in evaluating end-user's utility.

Form factor [6] measures the physical size and configuration of a machine, computer, mobile phone etc. Some such measures include size of a motherboard, display etc in PCs, size of computer peripherals, etc. This is a major factor for the overall user-friendliness of any computing or communication device. However, we consider this factor is beyond the scope of our study as we are comparing the utility of an office application in a PC environment. Form factor will play a key role in the comparison of PC vis-à-vis mobile environment.

GOMS [7] measures the usability of a software application from an end-user's perspective by estimating the execution time. The execution time is calculated by describing a task and then defining this task into four aspects: Goals, Operators, Methods and Selection rules, from a user's perspective. The end-user is considered to be skilled. Advanced versions of GOMS exist such as GOMS-KLM, NGOMSL, CPM-GOMS.

We considered the simple version of GOMS, GOMS-KLM due to its simplicity and shorter learning period which was suitable for our purpose. The KLM method specifies certain time for each of the operators identified such as clicking the mouse, pressing the keys, menu buttons etc. However, there are some limitations in this model which is explained later in this report.

Various tools have been developed for building GOMS-based models. Comparison of some such tools are provided in [8]

While KLM has been largely used for an application in a PC environment, some recent research has been done to calculate the execution time for different operators in a mobile environment [9]. Such efforts are quite relevant to evaluate usability of software applications in a mobile phone.

Software bloat [10] is created as a result of an increase in new functionalities to a point where the benefits of these added functionalities are outweighed by the increase in technical resource requirements. This is an important aspect of utility. However, the correlation between resource requirements and the utility of an end-user is unclear and hence requires more research. Some aspect of this has been covered in our results and questionnaire in later sections of this report.

Other possible attributes for evaluating the user aspects of software applications include security, effectiveness, efficiency and aesthetics. However, much of these attributes are abstract and can be analyzed based on end-user surveys. We have included some of these aspects in our questionnaire.

3 The model

We have developed our model based on multiple attributes that represent the end-user utility and resource required. We have looked at utility of software applications mainly from the perspective of HCI. The attribute scores are calculated and normalized in order to achieve the utility and resource indices. The concept of additive utility is applied to generate the utility function and a similar approach is followed for the resource measurement.

3.1 Methodology

The methodologies used for our study involves:

- Literature survey
- Constructive approach:
    - **Model development for utility measurement**: No single model available in literature was satisfactory for our requirements. Hence, we our approach has been to develop a framework combining multiple attributes. Some of the attributes identified are as follows:
        - Usability
        - Menu count
        - Cost of the application
    - **Model development for resource measurement**: We measured the resource utilized by the application based on the following attributes.
        - RAM memory (MB)
        - Hard disk space (MB)
        - Processor speed (MHz)
    - **Questionnaire development and analysis**: we conducted an end-user survey (sample space of 60). Opinions-online platform [11] was used for this purpose.

Appropriate weights were assigned for the attributes in each of the above mentioned measurements using SMARTER [12].

- Customer segment considered for our study **is business customer** as Office applications are mostly used by this segment. For the end-user survey, student community at HUT constituted the majority of the sample.

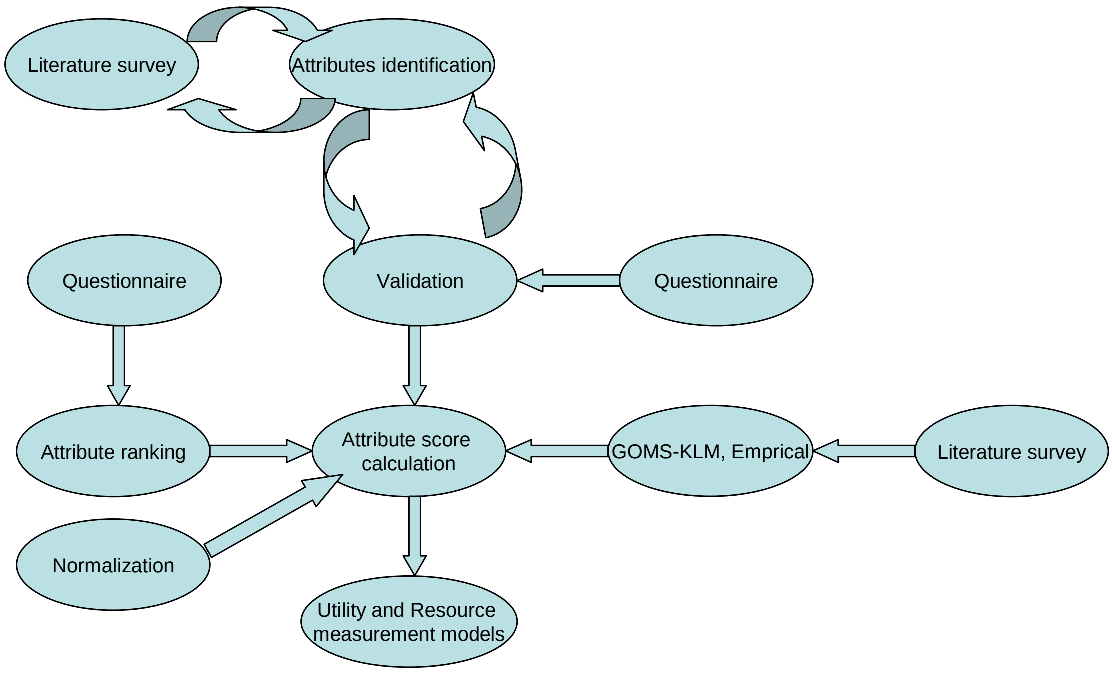


Figure 1 The methodology

3.2 Model calculation procedure

As mentioned before, we have adopted the additive utility concept for our calculation of utility index and a similar approach is adopted for resource index calculations. The following equations illustrate our model.

For Utility index:

$$U = w_1U_{usability} + w_2U_{menucount} - w_3U_{cost}$$

For Resource index:

$$R = w_1R_{memory} + w_2R_{diskspace} + w_3R_{processorspeed}$$

The weights for each of the attributes in the above equations are calculated using SMARTER technique. SMARTER's weight equation is given as follows:

$w_i = [2(n + 1 - R_i)]/ [n(n+1)]$ where, w_i = weight of the i^{th} attribute
R_i = Rank of the i^{th} attribute
n = Total number of attributes

The normalization of each of the attribute scores for applications were conducted by dividing the individual scores of each of the application for that attribute with the sum total of attribute scores of all the applications considered in a certain case.

However, there was an additional step required for calculating the attribute scores for usability. This was mainly required to convert the sum total of execution times, obtained from the basic functions calculated using the KLM model, to attribute scores that represent an increasing level of usability with decreasing execution time. This was achieved by subtracting the total execution time from 100 before doing the normalization as explained before.

3.3 Utility attributes

The utility attributes identified are as follows:

1. Usability

This attribute measures an end-user's experience in using an application in terms of the execution time taken for a certain task or function. We adopted GOMS model [7] for the evaluation of usability. A set of most commonly used functions were identified and a simplified version of GOMS, Keystroke-Level Model (KLM) [13] was used to estimate the execution time for each function. The functions identified for our analysis are as follows:

- **a.** Save As (into different formats)
- **b.** Print a file (pages 2-5, one-sided)
- **c.** Edit text
  - **i.** Style and formatting
  - **ii.** Replace text
- **d.** Insert table of contents
- **e.** Insert pictures from a file
- **f.** Adding equations
- **g.** Launching a web-link from the document
- **h.** Using the help options to search text

These functions cover the basic procedures of text processing. To obtain unbiased, comparable execution times the functions needed to be described very strictly. Without proper harmonization of the functions there would have been unnecessary variations in execution times. For example, printing a file was described as "print pages 2-5, one-sided" and replacing text was instructed to be performed as follows: "replace all, do not close the window after the operation, do not include any acknowledgements like '12 occurrences replaced, click OK to continue' into the count."

An example for the calculation of execution time using KLM is as follows:

Goal: To save a file into different formats
Application: MS Word 2002
Execution steps:

- Point to *File*	**P**
- Click	**BB**
- Point to *Save As*	**P**
- Click	**BB**

- Home hands to keyboard
- Type in filename (20 chars) **T(20)**
- Home hands back to mouse **H**
- Point to File format drop menu **P**
- Click **BB**
- Point to desired file format **P**
- Click **BB**
- Point to *Save* **P**
- Click **BB**

Total execution time (based on KLM):

5 * **P** + 5 * **BB** + 2 * **H** + **T(20)** = **12,9 sec**

2. Menu Count

While usability attribute provides measurement for the most common functions in an application, this is not sufficient to estimate the complete set of functionalities. Menu count provides the maximum number of functions available in an application. We have conducted the menu count manually up to two levels of menu items.

3. Cost

The amount of money (total cost of ownership) incurred on an application can be an additional factor that influences a customer's decision to use an application.
The sub-attributes considered for cost are as follows:

a. Purchasing cost
b. Maintenance cost (includes cost for upgrades)
c. Personnel cost (includes the cost of training/re-training as the versions change)
d. Switching cost
    - Switching cost increases with greater network externality.

Since we are focusing on the business users where cost may not be a major issue, this attribute was not considered in our studies. Moreover, we are looking at utility of an application after the purchasing decisions are made and hence the cost attribute's influence, if any, will not be applicable at this stage.

3.4 Resource attributes

In order to measure the resource requirements, we have identified the following sub-attributes.

a. RAM memory (MB)
b. Hard disk space (MB)
c. Processor speed (MHz)

The values we are using are recommended values by manufacturers. Minimum requirements might be smaller.

3.5 Limitations of the model

The attributes considered in our model may not be comprehensive. This means that there may be other attributes that can either be added or replace the existing attributes to increase the accuracy of utility and resource calculations.

We had used the simplified version of GOMS, i.,e KLM due to our time constraints. However, there are other sophisticated models of GOMS as mentioned in Chapter 2 that will enhance the quality of results.

The questionnaire sample space is limited to 60 and many of the participants are students from HUT. Since, we were looking at business customers, these results don't correctly represent the customer segment. Hence, a more accurate and elaborate survey should be conducted to verify and improve upon our results.

4 Results and analysis

This section summarizes some of the results achieved by applying the model in our studies on word processor applications. We considered two cases in this phase for our analysis. In this report we have focused our attention on usability, menu count and resource comparisons. Our analysis is based on the assumption that the purchasing decision has already been made by the end-user. Hence, cost is not considered as an influencing factor. The ranking of different attributes for utility and resource requirements were mainly based on our own assumptions and validated by the questionnaire responses.

4.1 Case 1: Comparison of different versions of MS Word

Three different versions of Microsoft Word were analyzed. The versions are MS Word 2.0 for Windows, MS Word 97 and MS Word 2002 from years 1992, 1997 and 2002 respectively. These three versions provide us with a time span of 10 years, 5 years between each version.

The hypothesis is that despite the large increase in resource requirements the overall utility has not increased respectively.

4.1.1 Utility

It seems that there are very minor changes in execution times between different versions of MS Word (Table 1). In some cases the execution times have even increased slightly. One reason for this could be that when the number of features (menu items) has increased the user interface has needed to be re-designed for clarity. This has produced deeper levels into the user interface and the user has to do more work to find the feature she is looking for. The feature is not immediately available in the first menu anymore. Illustrative examples of this development are accessing style and formatting and inserting table of contents:

In MS Word 2.0 the user could insert a table of contents directly from the Insert menu (a first-level menu item) but in MS Word 2002 the user has to dig deeper into the process of inserting different types of tables and indexes. In MS Word 2.0 there was only one kind of table of contents but in later versions the user may choose from larger variety of layouts.

Modifying paragraph settings was quite directly available for the user in MS Word 2.0 but in MS Word 2002 the user has to first open a Styles and Formatting panel and from that panel choose the desired style and then click a few more times to get to modify paragraph settings. However, the panel remains open (if so desired) and modifying paragraph settings will be slightly faster afterwards.

Small changes in execution times indicate that these actions are considered to be of very basic level and commonly used by all users. Microsoft has not wanted to change the way these basic actions are performed in order to maintain the familiarity of the user interface. As revealed by our questionnaire, switching costs are widely considered to affect the selection of a word processor.

Table 1 Execution times of selected actions in three different versions of MS Word. There are no large differences between these versions but surprisingly in some tasks new versions seem to be slower than their antecedents.

	MS Word 2.0 (year 1992)	**MS Word 97**	**MS Word 2002**
Save As	12,9	12,9	12,9
Print	6,84	6,84	6,84
Style and Formatting	5,2	6,5	7,8
Replace	10,58	10,58	10,58
Table of Contents	3,9	5,2	6,7
Insert Picture	5,2	6,7	6,7
Add equations ($f(x)=x^2$)	15,06	15,06	15,06
Launching weblinks	N/A (3)	1,3	1,98
Help options	3,9	3,9	2,6
Total (sec)	**66,58**	**68,98**	**71,16**

All execution times were summed up and based on the sum scores were given for each application. Table 2 illustrates how these scores have been calculated. The same was applied to menu count as well, thereby obtaining normalized scores.

Table 2 Normalized scores for usability and menu count

	MS Word 2.0	MS Word 97	MS Word 2002
Total (sec)	***66,58***	***68,98***	***71,16***
100-total	*33,42*	*31,02*	*28,84*
Normalized score	*0,36*	*0,33*	*0,31*
Menu count	**93**	***154***	***178***
Normalized score	*0,22*	*0,36*	*0,42*

Finally, total utility was derived from usability and menu count. Weights were calculated using SMARTER method (Simple Multi-Attribute Rating Technique Exploiting Ranks). Usability was considered more important than menu count, thus producing weights 0,67 and 0,33 for usability and menu count respectively.

Table 3 Utility derived from usability (KLM times) and menu count

	Rank	*Weight*	**Word 2.0**	**Word 97**	**Word 2002**
Usability (KLM)	1	0,67	0,24	0,22	0,21
Menu count	2	0,33	0,07	0,12	0,14
Total Utility			**0,31**	**0,34**	**0,35**

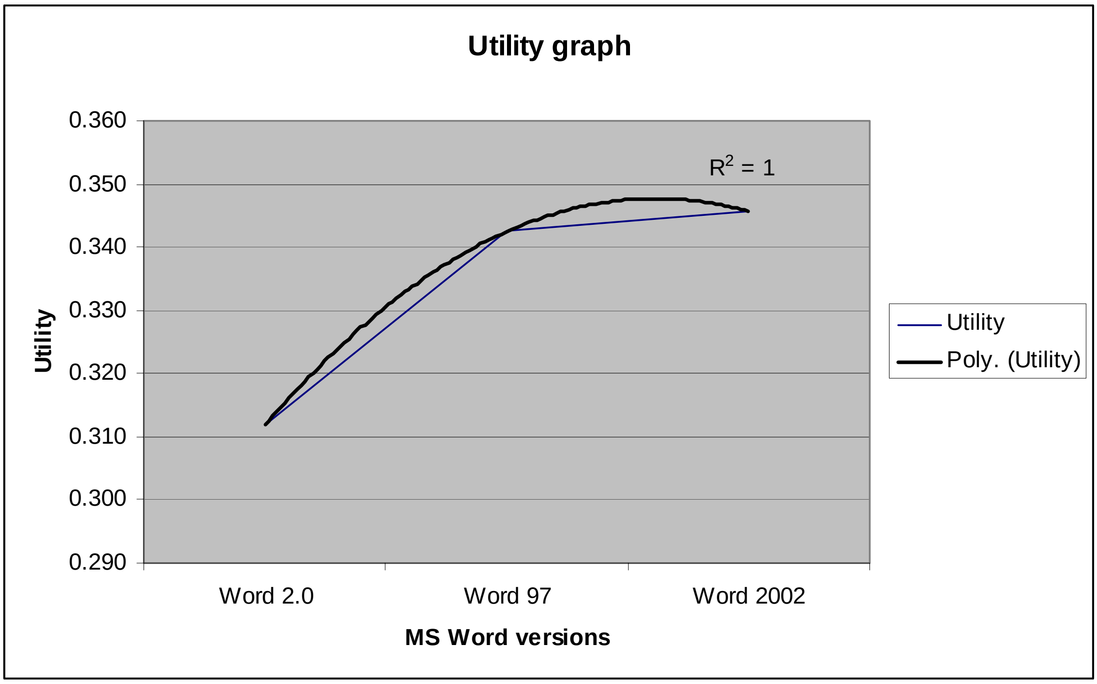


Figure 2 Development of utility over time. From 1992 (Word 2.0) to 2002 (Word 2002) the utility has increased slightly. The shape of the utility curve seems to be concave and the increased utility in the latest versions has been smaller than in the first versions.

The utility has increased over time, but the utility curve is concave (Figure 2). This might be a coincidence caused by the limited number of data points. However, the concavity of the utility curve stems with our hypothesis: added value to the user has not increased as rapidly during the recent years as it did when word processors for MS Windows were first introduced in the early 1990s. The trend line is polynomial of order 2, with R^2 equal to 1.

The trend line based on our data if extrapolated further will show a decrease. However, this result is not considered as conclusive due to the limited amount of data involved in this analysis and **we recommend further work on this aspect.**

Added value between older and newer versions of MS Word does not seem to emerge from easier use of these basic functions. More probably only the overall number of features could provide the users with an incentive to upgrade their MS Word versions. MS Word 97 has 154 menu items (first and second level) while MS Word 2002 has 178 menu items. MS Word 2.0 for Windows has only 93 menu items, all of them being first-level menu items. Thus, the overall number of functions available to the user has increased. If the user does not need these new features or if adding those new features increases resource requirements to excess, choosing an older version of MS Word could be recommended for the user. Smaller number of functions is also recommended when the platform where the application is run offers only limited resources.

4.1.2 Resource requirements

During the ten years resource requirements have increased exponentially. While MS Word 2.0 for Windows run on a 80286 computer with 2 MB of RAM and 6 MB of free disk space, ten years later MS Word 2002 is recommended to be run on a Pentium III system with 128 MB of RAM and 265 MB of disk space. System requirements are shown in Table 4. These requirements are recommended configurations for systems that are able to run the applications smoothly (as provided by Microsoft in [14], [15], [16]). Minimum requirements might be smaller. Processor speeds (MHz) are speeds of typical 286, 486 and Pentium III processors.

Table 4 Resource requirements

	Recommended Processor	Processor (Mhz)	Memory (RAM, MB)	Disk Space (MB)	Total resource requirement
Word 2.0	286	10	2	6[1]	**6,64**
Word 97	486	33	8	46	**35,25**
Word 2002	Pentium III	600	128	265	**352,26**
Rank		*2*	*3*	*1*	
Weight		*0,33*	*0,17*	*0,50*	

[1] Approximately 6 MB, based on installation size of AMD Athlon 700 MHz, 20 GB, 256 MB. Varies between 5-15 MB according to installed packages. [15]

Again, weights were calculated with MS Excel using SMARTER. In the column Total are weighted sums of processor speed, memory requirement and free disk space requirement.

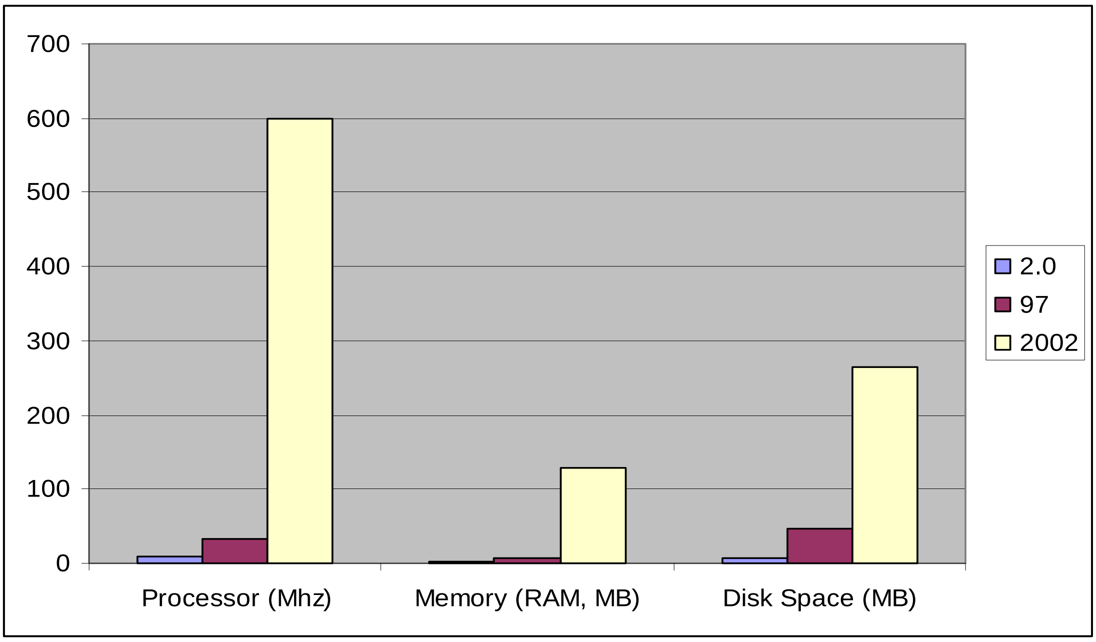


Figure 3 System requirements for MS Word versions in graphic form

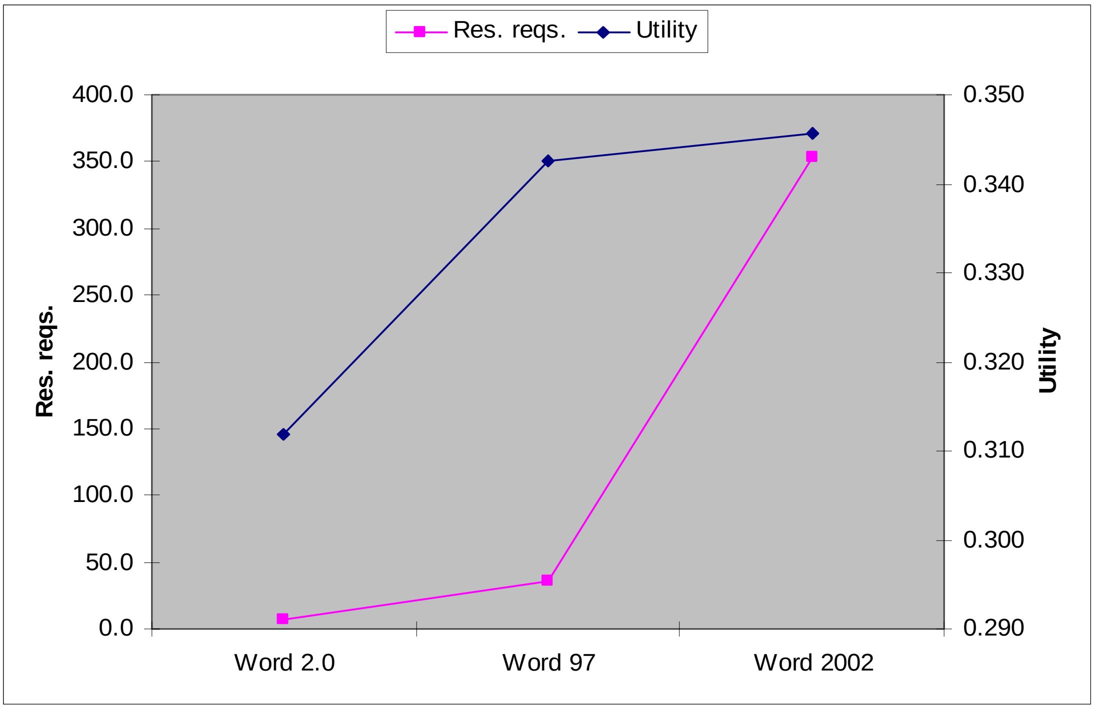


Figure 4 Resource requirements versus utility. Resource requirements have increased exponentially hand in hand with home PCs computing capabilities. Utility has not increased with resource requirements.

Figure 3 illustrates the same development. As we see, system requirements have increased dramatically over time. However, the development is not as alarming as the striking shape of the figures could suggest because the performance of computers have increased rapidly as well. At the time of their release, the current versions of MS Word have always run finely a state-of-the-art PC of their time.

However, increased resource requirements may cause significant problems in environments where the platform has limited computing capabilities. This applies especially to hand-held devices such as mobile phones and PDAs. In hand-held platforms processing power is limited due to heat production and electricity usage issues. Constrained processing power places limitations to the applications respectively.

Table 5 Utility-Resource ratio

Applications	Utility (U)	Resource required (R)	Utility-resource ratio (U/R)	Menu count
Word 2.0	0.31	6.64	0.046687	93
Word 97	0.34	35.25	0.009645	154
Word 2002	0.35	352.26	0.000994	178

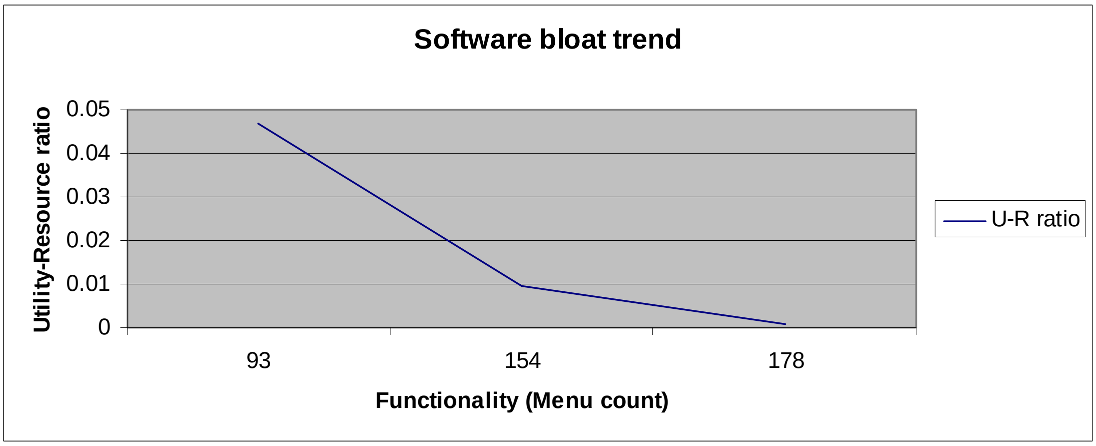


Figure 5 Software bloat trend

Table 5 and Figure 5 shows that the phenomenon of software bloat exists. However, considering the limitations in our model, **we would recommend further work on this aspect.**

The conclusion is that despite limited resources it is still possible to offer a sound word processing application with all relevant features by cutting down more sparsely used functionalities.

4.2 Case 2: Comparison of different word processor applications

In this case, we compared word processors offered by different providers. In order to avoid any discrepancy in comparison, all measurements were conducted on the same **notebook computer (HP Omnibook XE3) with Windows XP OS, 256 MB RAM and a Pentium 3 processor.**

4.2.1 Utility

Utilities are calculated in the same way as in the first case. Based on total execution times (Table 6) and menu counts OpenOffice has the largest utility. AbiWord comes second and MS Word has the smallest utility. Total utilities are shown in Table 8.

AbiWord and MS Word have similar menu counts which imply that their overall numbers of features should be on the same level. OpenOffice's menu count is significantly larger suggesting more features available for the user. However, due to limitations of the menu count model the real amount of features might differ from these values; menu count is a rough approximation.

Table 6 Functional comparison of different word processor applications

Functions	MS Word 2002	Open office(v1.1.2)	AbiWord
Save	12,9	14,2	13,8
Print	6,84	11,24	6,84
Style and Formatting	7,8	2,6	6,3
Replace	10,58	12,16	10,58
Table of Contents	6,7	5	2,6
Insert Picture	6,7	8,7	5,2
Add equations ($f(x)=x^2$)	15,06	9,72	N/A(16)
Launching weblinks	1,98	1,3	N/A (3)
Help options	2,6	2,6	2,6
Total	71,16	67,52	67,22

Table 7 Normalized scores for usability and menu count

	MS Word 2002	***OpenOffice(v1.1.2)***	***AbiWord***
Total (sec)	***71.16***	***67,52***	***67,22***
100-total	*28,84*	*32,48*	*32,78*
Normalized score	*0,306482*	*0,345165*	*0,348353*
Menu count	**178**	**214**	**168**
Normalized score	*0.317857*	*0.382143*	*0.3*

Table 8 Utility derived from usability (KLM times) and menu count

	Rank	*Weight*	**Word 2002**	**OpenOffice(v1.1.2)**	**AbiWord**
Usability (KLM)	1	0,67	0,21	0.23	0.23
Menu count	2	0,33	0,10	0.13	0.10
Total Utility			**0,31**	**0.36**	**0.33**

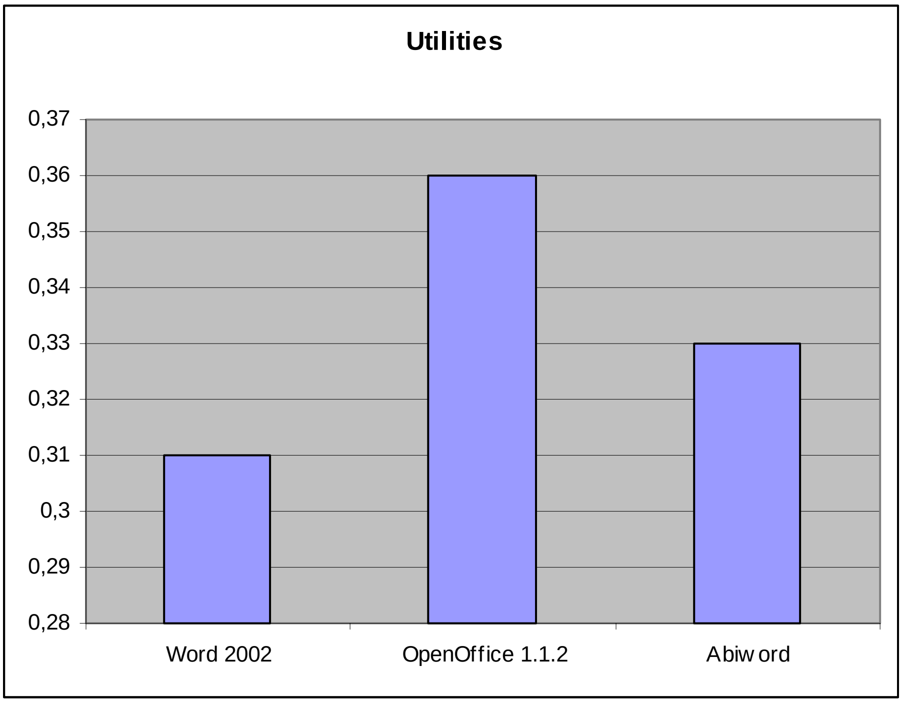


Figure 6 Utilities of MS Word, OpenOffice and AbiWord

4.2.2 Resource requirements

MS Word 2002 has clearly the largest resource requirements (Table 9). OpenOffice has moderate resource requirements and runs without problems even on an older computer. AbiWord is the lightest of these word processors, it has very minimal resource requirements and runs even on a 486 computer with 16 MB of RAM. When comparing the disk space requirements it has to be noted that OpenOffice is a full office application and includes e.g. a spreadsheet application while AbiWord is only a word processor. This explains some of the differences in disk space requirements.

Table 9 Resource comparison

	Processor (type)	Processor (Mhz)	Memory (RAM, MB)	Disk Space (MB)	**Total resource requirement**
Word 2002	Pentium III	600	128	265	**352.26**
OpenOffice 1.1.2	Pentium	150	64	250	**185.38**
Abiword	486	33	16	45	**36.11**
Rank		2	3	1	
Weight		0.33	0.17	0.5	

Figure 7 illustrates the large differences in resource requirements graphically.

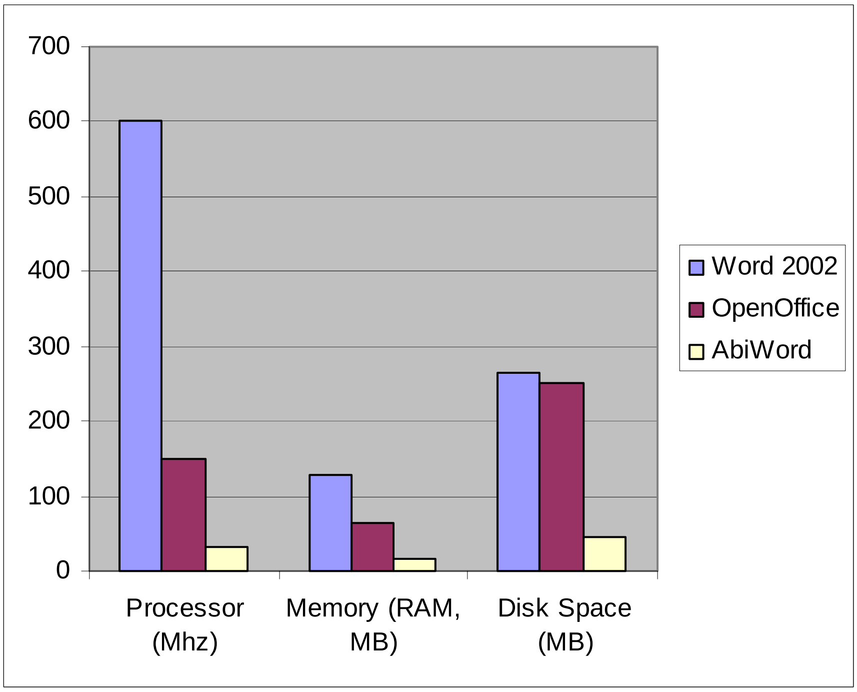


Figure 7 Resource requirements. MS Word places the largest system requirements.

When the utility gained by the user is plotted with resource requirements it is revealed that even though MS Word 2002 has the largest demands on the system it still does not provide the largest utility (Figure 8). Actually when utility is divided by resource requirements AbiWord has the best ratio. It is able to provide the user with all significant features and still require a reasonable amount of computing capabilities.

When designing office applications on a platform with limited resources one should aim to maximal utility-resource ratio to best use the limited computing capabilities.

Table 10 Utility-resource ratios for MS Word, OpenOffice and AbiWord

Applications	Utility (U)	Resource required (R)	Utility-resource ratio (U/R)	Menu count
MS Word 2002	0.31	352.26	0.000880032	178
OpenOffice	0.36	185.38	0.001941957	214
AbiWord	0.33	36.11	0.009138743	168

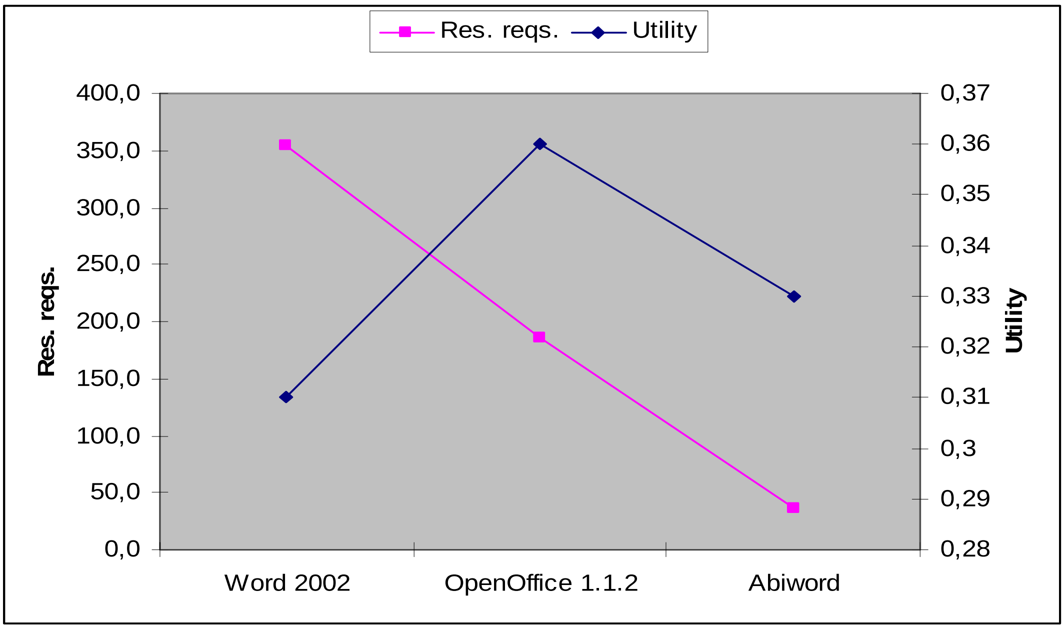


Figure 8 Utilities and resource requirements in a same graph

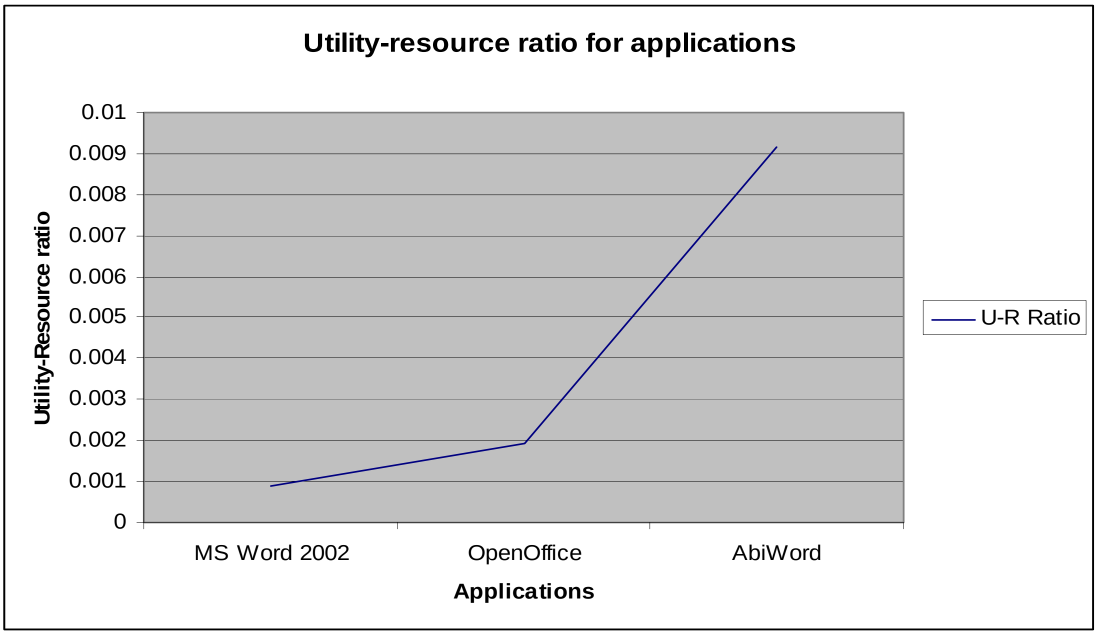


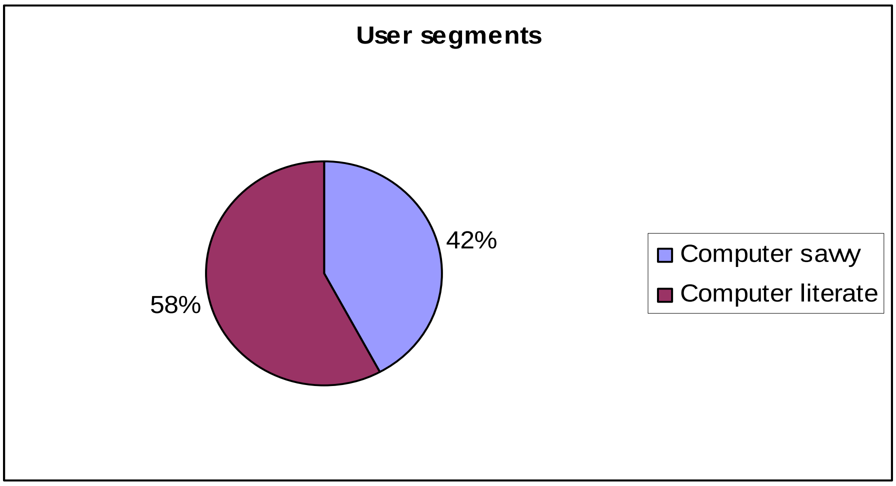
Figure 9 Utility-resource ratio for different office applications

Our analysis shows that AbiWord has better overall utility-resource ratio over the other two applications. It is worth mentioning in this regard that that all three applications have differences in their design structure. While MS Word is a separate module in the MS Office suit, OpenOffice appears to have a monolithic design and AbiWord is a standalone word processor application.

4.3 Questionnaire

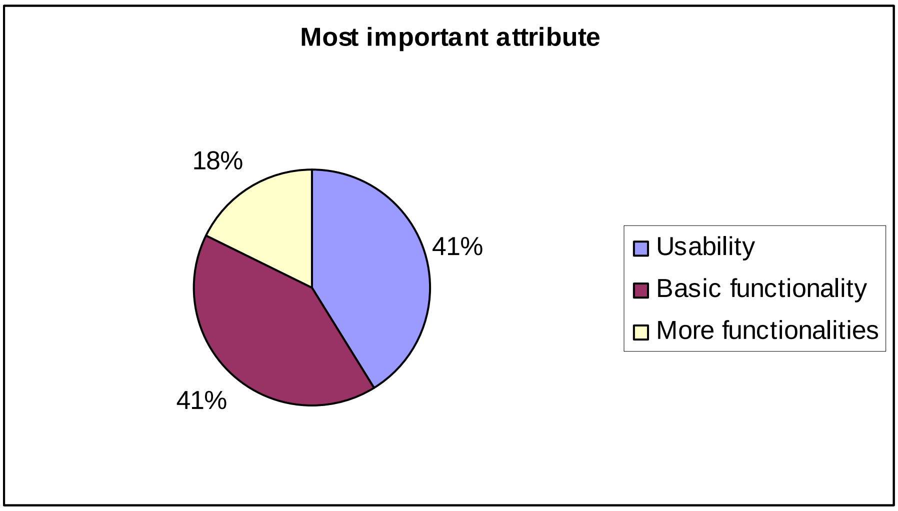
Based on the questionnaire prepared by us (refer Appendix A for the questions), we received the following results from a sample space of 60, mostly students from HUT. The questionnaire was launched online on the Opinions-online platform [11]. The results are based on a sample space of 60, mostly students from HUT, over a period of one week. The questionnaire helped us to validate some of our assumptions and also threw up some interesting and some expected results.

I. User segment?



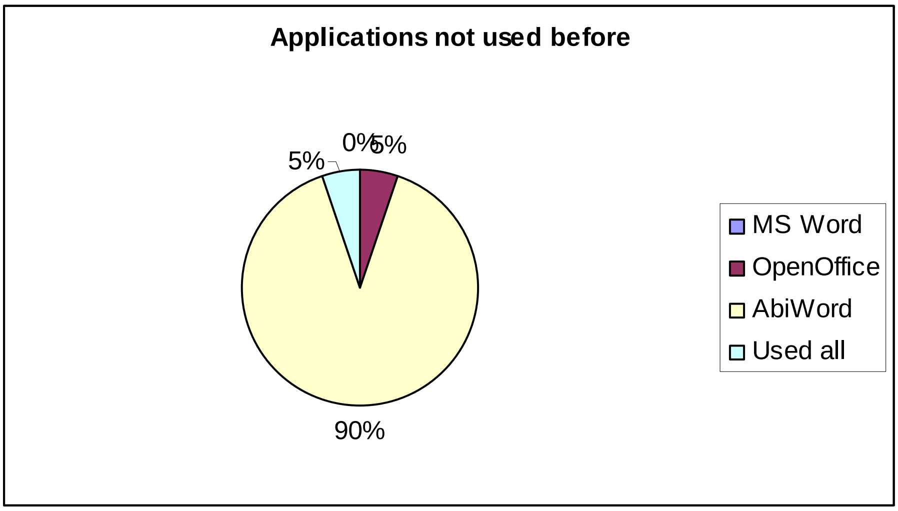
First of all, we wanted to know how the answerers of our questionnaire are divided in terms of user-specific preferences. 42% of them felt that their user-specific behaviour was more into complexity-loving, efficiency-looking manner, and the rest, 58%, felt that they are content with functions that satisfy their basic need of word processing

II. Most important attribute?



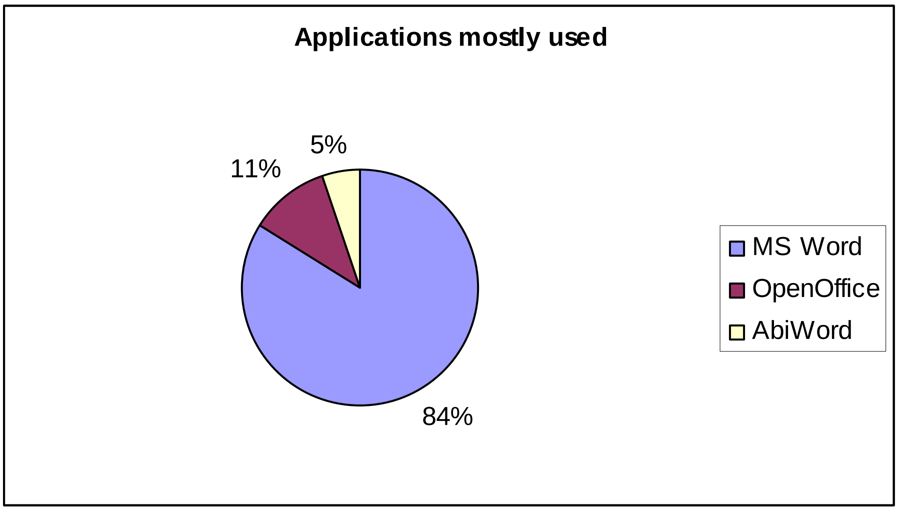
Next, we wanted to find out what attributes do users prefer in word processors. These attributes were divided into basic, more and usability attributes. As suggested by the previous question, most users did not prefer the ultimate number of functionalities in a WP, but only the basic functionalities or the feature of ultimate usability.

III. Applications not used before?



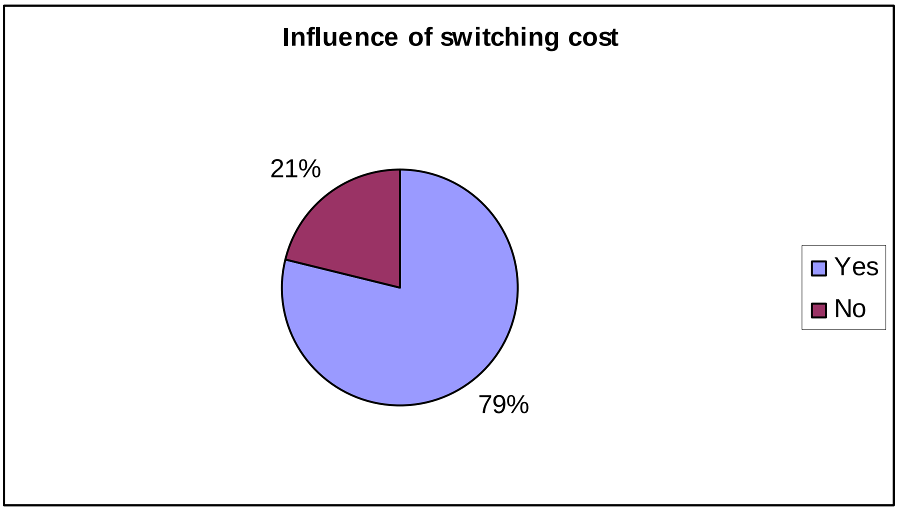
The arrangement of this question was only a technical one; we simply wanted to know about how the users knew previously the case applications (MS Word, OpenOffice, AbiWord) in our study. The result was not a surprise; almost all answerers did not know about AbiWord previously, and OpenOffice came second with a 5% share. Another 5% had used them all.

IV. Applications mostly used?



Our next question was about preferences. Due to the popularity and the market share of Microsoft programs, MS Word was the most preferred, with 84% share[2]. Next came OpenOffice and AbiWord, in 11% and 5% shares, respectively.

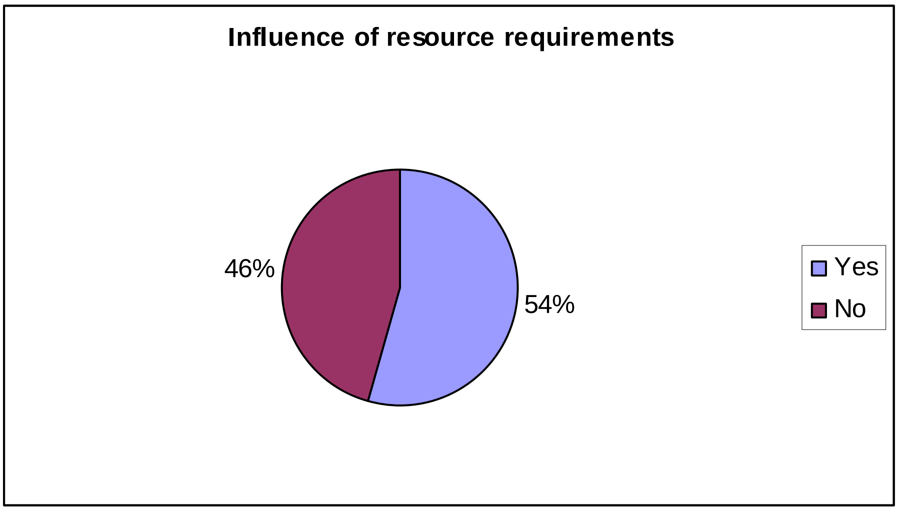
V. Influence of switching cost (incompatibility issues) on the choice?



In this question we wanted to know briefly that whether the switching costs influence on the choice of the specific application. Most users thought that it does.

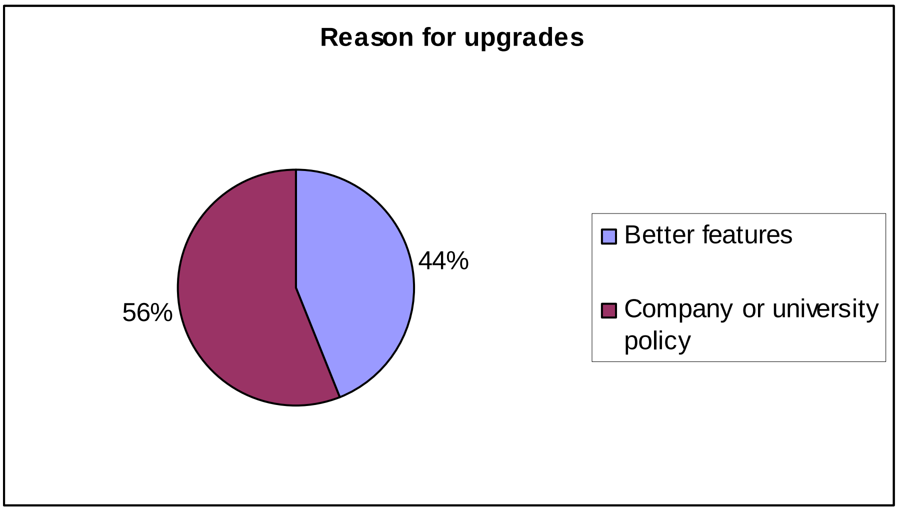
[2] Consulting and analysis company Gartner estimates that StarOffice, the main rival of MS Office, has about 20% market share. This number seems to fit well in our result.

VI. Influence of resource requirements (start-up time, CPU usage) on the choice?



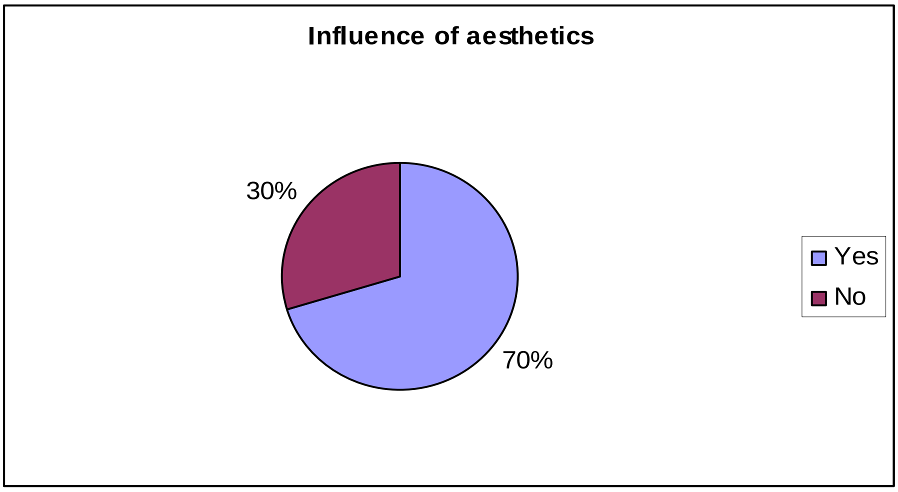
In the PC world, one might not consider the resource requirements nowadays, as there are affordable hardware available with enough resources to run even the most sophisticated applications. Although you might assume that when a application manufacturer promises that the program will run effectively in such and such environment, you still get to wait long for some program to start. In our consideration, OpenOffice was far too slow to start (see previous chapters). The results of this question were not so useful because of the even division of opinions.

VII. Reason for upgrades ?



A common feature of computer software is the fast pace of upgrading. Usually the reasons for upgrading are not so obvious. You might want to upgrade because the newer version has better features that the older, or you might *have to* upgrade because you are told to do so or someone else does it for you. This question also had even-deviated result, but since our target group was specifically the student community and the business users, it is obvious that these users use mostly word processors that someone else has set up for them, i.e. the upgrading decision is included in the company or university policy.

VIII. Influence of aesthetics (nice looks) on the choice?



An important attribute of the user interface of a computer application is that it is aesthetically pleasant. 70% of our target group agreed that the nice looks of a word processing application has an effect on the purchasing decision. This question was commented by the claim that one might not know how the application looks before the purchasing decision. We also requested the answerers to select the most aesthetic one of the three screenshots of our case applications. MS Word was the winner of this, with a bit over 50% of the votes, and OpenOffice and AbiWord came second and third with almost even amount of votes.

4.4 Document analysis

One way to reveal differences in the usability of word processors could be analyzing large documents instead of single tasks. This would hopefully amplify the slight differences between word processors' KLM execution times.

However, this approach may not necessarily produce the desired outcome. When large documents are written by human beings the writing process does not always run fluently and consistently. This writing process is not simple and unambiguous; people make mistakes and sometimes they use different ways to perform tasks. All this adds "random noise" to overall execution times and conceals the real differences between word processors.

Simply scanning through a document mechanically and counting pictures, tables and other elements it includes and multiplying those counts by their respective KLM execution times would not change the overall ranking of the word processors. Taking two execution times, smaller and larger, and multiplying them by some integer would not change their order.

Thus, we decided to drop this analysis. The only benefit this approach could provide could be a performance comparison of different word processors when producing different types of documents, e.g. legal documents or mathematical publications with large number of equations. In different types of documents the number and type of the elements they include varies, which might reveal that some word processor could be better when writing legal documents while some other application would be better when writing documents with lots of complex equations.

What is especially problematic about KLM approach is that the usage of these average execution times for certain procedures might result in the loss of some essential information concerning the typing process itself. Assuming that different types of documents (legal, technical, etc.) may require different types of features and usability from the user, a thorough analysis would cover the varying characteristics of them. However, since we do not possess this kind of knowledge, we are not fully capable of analyzing the actual usability with respect to the application area. Simply analyzing an existing, say legal document, would not give correct results.

For future studies it would be perhaps very beneficial to try somehow to obtain area-specific behavioral factors in to the focus of the analysis. This would significantly increase the amount of relevant data for software development.

4.5 Effort required

Performing KLM calculations for the given set of basic function identified in our study is not very time-consuming. For one application this can be done in about half an hour. However, the key here is to identify the list of basic functions and harmonize the method used to calculate the execution time for each function. This is crucial to ensure accuracy of the model. The identification of the list of functions was carried out among the team members by collecting individual list of basic functions used and subsequently finalizing a common set of 9 functions. The overall effort required a series of discussions for over a week. It’s worth mentioning here that a more elaborate method is to identify the list through questionnaires. Doing the menu count takes only one minute. Increasing the number of applications would not have significantly increased the effort required in that sense.

Quest for older versions of MS Word was rather inconvenient and required some time and effort. It seems that installation disks of older applications are normally thrown away.

5 Conclusions

The major contributions of this study are 1) the development of utility and resource requirement models 2) additional insight to some of the research questions through a questionnaire survey and 3) a reference document for further studies in this domain. Based on our literature survey, we realized that the concept of utility and resource measurement related to a software application has not been tackled comprehensively. In other words, different attributes has been dealt with separately, but an effort to identify and combine these attributes to calculate utility and resource requirement values is missing. Our study provides an introduction to such an effort. The identified attributes are not comprehensive in every sense and would require further work and analysis. However, we hope that the methodology developed in this study will be beneficial for any future extension of this work.

The results achieved by the application of our model have highlighted some interesting trends that require deeper analysis. The relative values of utility for word processors in the two cases are some such results. Our study also shows the existence of software bloats. If this is true, then it proves that the increase in new functionalities have resulted in an increase in resource requirements. This is an important result, especially for mobile phone and application vendors, and requires further studies. The result is particularly interesting in light of our questionnaire result that shows that only 18% of the participants are interested in advanced or new functionalities. The correlation between utility and resource requirements should also be looked at closely.

From a mobile perspective, where the handset is more personal to the user, cost and form factor are some of the additional attributes that may have an impact on the utility for an end-user.

Any work in the future should consider the limitations of our model described in this study.

Acknowledgement

We express our sincere gratitude to Prof. Ahti Salo (Systems Analysis Laboratory, HUT) for his support and guidance throughout the course of this study.

We thank Jukka K. Nurminen (Principal Scientist, Nokia Research Centre) for giving us an opportunity to conduct this study and for his timely advice.

We very much appreciate the support by Juuso I. Liesiö (course assistant) for facilitating communications related to the course.

Last but not the least, we thank all the anonymous people who participated in our online questionnaire survey without which this work would have been incomplete.

Reference

[1] Moore's Law, http://www.intel.com/research/silicon/mooreslaw.htm
[2] Robert T. Clemen, "Making Hard Decisions-An introduction to decision analysis", 2nd edition, Duxbury, Cambridge, 1996.
[3] Function Points, http://www.functionpoints.com/default.asp
[4] IFPUG, http://www.ifpug.org/
[5] Feature Points, http://www.spr.com/products/feature.shtm
[6] Form factor, http://www.formfactors.org/
[7] Bonnie E. John, "Why GOMS?", ACM Interactions Magazine, Vol. 2, Issue 4, October 1995.
[8] Baumeister L.K., John B.E., Byrne M.D, " A comparison of tools for building GOMS models", Proceedings of the SIGCHI conference on Human factors in computing systems, http://portal.acm.org/citation.cfm?id=332040.332485
[9] Mori R., Matsonobe T., Yamaoka T., "A task operation prediction time computation based on GOMS-KLM improved for the cellular phone and the verification of that validity", http://www.idemployee.id.tue.nl/g.w.m.rauterberg/conferences/CD_doNotOpen/ADC/final_paper/357.pdf
[10] McGrenere, J.,Moore, G. "Are we all in the same "bloat"?" *Proceedings of Graphics Interface 2000*, 187-196, http://www.cs.ubc.ca/~joanna/papers/GI2000_McGrenere_Bloat.pdf
[11] Opinions-online, http://www.opinion.hut.fi/
[12] SMARTER technique, http://www.hipre.hut.fi/WebHipre/Help-Priorities.html#Smarter
[13] Kieras D., "Using the Keystroke-Level Model to estimate Execution Times", ftp://www.eecs.umich.edu/people/kieras/GOMS/KLM.pdf
[14] MS Word 2002 system requirements, http://www.microsoft.com/office/previous/word/2002sysreqs.asp
[15] MS Word 2.0 system requirements, http://www.emsps.com/oldtools/mswordv.htm
[16] MS Word97 system requirements, http://support.microsoft.com/default.aspx?scid=kb;en-us;159417
[17] OpenOffice system requirements, http://www.openoffice.org/dev_docs/source/sys_reqs_11.html
[18] AbiWord system requirements, http://www.abisource.com/support/require/

Appendix

Appendix A: Questionnaire

The following is the questionnaire used for our studies.

Utility assessment of Office software applications (such as word processors)

All answers are handled anonymously.

Which option best describes your computer application usage behaviour?

- [ ] Computer savvy. Loves complexities. Creates documents with advanced functionalities. Always uses efficient methods for usage.
- [ ] Computer literate. Likes to avoid complexity. Creates documents with simple functionalities. Does not mind inefficient usage.

Check the most important attribute you look for in a word processor?

- [ ] Usability (easy to use)
- [ ] Basic functionalities (good enough to satisfy the basic needs)
- [ ] More functionalities (more the merrier)

Which of the following processors have you <u>not</u> used before?

- [ ] MS Word
- [ ] OpenOffice
- [ ] AbiWord
- [ ] I have used them all
- [ ] I haven't used any of them

Which of the following applications do you like to use the most?

- [ ] MS Word
- [ ] OpenOffice
- [ ] AbiWord

Does switching cost (incompatibility issues) influence your decision to purchase?

- [ ] Yes
- [ ] No

Does resource requirements (start-up time, CPU usage) influence your decision to purchase?

Yes

No

What is the reason for your upgrading to newer versions?

The newer version has better features

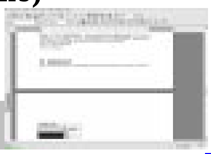
Because I have to (company or university policy)

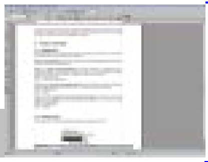
Does aesthetics (nice looks) play any role in your choice of a word processor?

Yes

No

Here you can see screenshots of three different word processors (click to make larger). Select the one that you find the most aesthetic (nicest looks)





Additional comments/opinions:

List 3 best and worst functions you find in a word processor? (Voit myös vastata suomeksi)

Submit

Appendix B: Questionnaire Comments

Finally, we asked each respondent to name three best and three worst functions that they find in a word processor. We received some 30 replies for this.

The most popular functions that were named in general (regardless of the application) were the good structure management, easiness of use associated with quick menus and the clarity of the user interface. The functions that would make any WP bad were automatic text editing, the non-visibility of the most needed functions and the unreliability or instability of the application.

Application specific opinions were heavily divided into two camps under some functions, such as spelling/grammar functions in MS Word. Most respondents hated that the application makes automatic corrections, or generally draws a red line under the misspelled text. There were some respondents that thought it is a good function, but needs enhancements, especially because the Finnish language is not the easiest for a computer to understand. Some respondents thought that spelling corrections was the best function in a WP.

MS Word was especially applauded for such functions as style management with "Styles and Formatting", making table of contents easily, easiness to insert objects and tables and the clarity of its user interface. The same application was panned for its incompatibility with other documents (such as .sxw), its overwhelming functionality and its unreliability. OpenOffice on the other hand, got respect for some functions that MS Word didn't, such as compatibility with other documents and its ability to convert to pdf. The equation editor of OpenOffice also got compliments. OpenOffice was panned because of its start-up time and heaviness.

There was a minor group of AbiWord users in our sample, so we did not get any comments about it directly.