DANISH BOARD OF TECHNOLOGY

Open-source software
- in e-government

Analysis and recommendations drawn up by a working group under the Danish Board of Technology

October 2002

Contents

Foreword 4
Summary 5
Recommendations of the working group 7
Introduction 8
Open-source software in e-government 9
1.1. Open-source software in e-government 9
1.2. E-government in Denmark 11
1.3 E-government according to the Digital Task Force 12
1.4 E-government in practice 12
1.5. Information Technology requirements of e-government 13
1.6. Why is open-source software of interest to the public sector? 14
1.7 Open-source software from the political point of view 15
1.8. Conclusions 15
What is open-source software? 16
2.1. Open source as practice and movement 16
2.2. Example: The Apache web server and its licence 17
2.3. The user's rights according to open-source licences 17
2.4. Distribution and payment 18
2.5. Open standards 18
2.6. Licences and copyright 19
2.7. Conclusions 20
Open source as desktop, infrastructure and custom built software 21
3.1. Definitions 21
3.2. Software on the workstation 21
3.3 Infrastructure software 25
3.4. Custom built software 29
3.5. Conclusions 29
Economic analyses of open source 30
4.1. Economic characteristics of software as a product on a market 30
4.2. Limitations of the analysis 31
4.3. The economic rights perspective 31
4.4. The economic development perspective 31
4.5. Options 32
4.6. Software costs in an options perspective 33
4.7. Upgrading in an options perspective 35
4.8. Licence ties 37
4.9. Economic model for an investment decision 38
4.10. Example of option 39
4.11. Conclusions 39
Economic analyses of the use of open source on the desktop 40
5.1. The general economic model 40
5.2. A fictitious example 41
5.3. Århus County 43
5.4. Hanstholm Local Authority 46
5.5. Copenhagen Labour Market Training (AMU) Centre 47
5.7. Compatibility 49
5.8. Market development 49
5.9. Overall assessment of the examples 50
5.10. Conclusions 51
Economic analyses of open source as infrastructure software 51
6.1. Linux vs. UNIX 53
6.2. Example: Danish Consumer Information Centre 59
6.3. Conclusions 61
Open-source and custom built software 62
7.1. Standard systems 62
7.2. Older systems and ownership of the source code 62
7.3. New development and ownership of the source code 62
7.4. One system, several user institutions 63

7.5. The need for development in e-government 63
7.6. The European Environment Agency 63
7.7. Three traditional scenarios for the development of new systems 64
7.8. A fourth scenario: Open source as a method of cooperation 64
7.9. Conclusions 66
The socio-economic consequences of open-source software 67
8.1. Qualitative socio-economic assessments 67
8.2. Quantitative socio-economic assessments 70
8.3. Conclusions 76
Conclusions and recommendations 77
Annexes 80
Glossary 80
MS Office/StarOffice document exchange test 82
Overview of Microsoft licences 85
Age of PCs that can be used in latest version of Office suites 87
Use and development of open-source software 89
Available reports on open-source software 91

Foreword

The Danish Board of Technology decided in 2001 to have a more detailed analysis made of the usability of open-source software in public administration, including the economic perspectives in a change-over to open-source software.

This report presents the results of this work. It contains an economic analysis of open-source software and additionally analyses its usability. The report also makes a number of recommendations on how the public sector can promote the use of open-source software.

The report has been drawn up by an interdisciplinary working group under the Danish Board of Technology. A number of people who together have special knowledge in the area and the necessary economic expertise were chosen as members of the working group.

The working group consisted of:

- **Jan Birk, Head of IT, Employment Training (AMU) Centre, Copenhagen**
- **Jens Hørlück, Associate Professor, School of Economics and Management, Aarhus University**
- **Niels Jørgensen, Associate Professor, Department of Computer Science, Roskilde University Centre**
- **Mogens Kühn Pedersen, Professor, Department of Informatics, Copenhagen Business School**

The Board of Technology made a secretarial office available for the working group.

The Board of Technology and the working group would like to thank everyone who has contributed to the project along the way and made data available: the Danish Commerce and Companies Agency, Microsoft, IBM and Mærsk IT. Special thanks are due to the public institutions which have provided the basis for the examples in the report: the Danish Consumer Information Centre, Århus County, Hanstholm Local Authority, the European Environment Agency and the AMU Centre Copenhagen. We wish to thank Simon Gjedde, Ministry of the Environment, and Thomas Myrup Kristensen, Ministry of Science, Technology and Development, for their contributions to the discussions of the working group.

Responsibility for the contents of the report rests solely with the working group.

Danish Board of Technology, October 2002
Jan Opstrup Poulsen, Project Manager

Summary

The conclusions of the report and the recommendations of the working group are based on the analyses in Chapter 1-8, the results of which can be summarised as follows.

Chapter 1 : Open-source software in e-government.
The public sector needs to change over to communicating digitally. This development makes great demands both on the IT systems on which e-government is based and on work processes in the public sector. From the economic point of view, the change-over poses great challenges, as huge investments will have to be made in IT in the public sector over the next few years. It is therefore natural, in connection with these investments, for detailed assessment to be made of what forms of information technology it is anticipated will be used, and who controls the development and ownership of this technology. The question is: to what extent open-source software can supplement or completely replace proprietary software.

Chapter 2: What is open-source software?
The core of the concept of open source is the user's access to the source code and right to alter and distribute the software which is laid down in an open source licence. Open source is also linked to attitudes on knowledge sharing, freedom and open standards.
The right to distribute means that the open-source software – but not if it is specially developed – can be obtained in return for the distribution costs, i.e. free or almost free of charge. Costs of adaptation, maintenance etc., which account for most of the total costs in connection with software, still have to be borne.

The right to make changes entails a right to choose a supplier for maintenance tasks, and the use of open standards by the open-source software provides greater freedom of choice with respect to the other software it is to be used in conjunction with.

Chapter 3: Open source as desktop, infrastructure and custom built software.
The prospects of increased application of open-source software on the desktop are heavily affected by the dominant position of Microsoft's closed file formats. The essential requirement to be met for increased application of open source on the desktop and for greater competition to be established in the area is for the public sector to make sure that word-processed documents are exchanged in an open file format.

The prospects of increased application of open-source software within the infrastructure software area are good, as the area is dominated by open standards and there are already high-quality established open-source products. High priority should be given to the issue of security in choosing between different options in the area of infrastructure.

In public calls for tender for work relating to custom built software, the public sector can demand that all or part of the software is supplied as open source. Consideration can also be given to demanding less far-reaching rights, such as access to the source code and the right to modify it.

Chapter 4: Economic analyses of open source.
The working group's economic analysis of the potential of open source looks at software as an option, and the analysis is based on two perspectives, the short-term rights perspective and the more long-term development perspective. The short-term rights perspective focuses on who owns the software, and what economic consequences this has for the user. When there are comparable products, the rights perspective identifies clear economic advantages in procuring open source rather than proprietary software. The long-term development perspective identifies important factors in relation to the development and maintenance of the software, where the use of open source generally requires greater local skill. Providers of proprietary software gain revenue through licence ties and through frequent upgrades. The costs of this can be reduced by using open-source software.

The ordinary market conditions for standard software will tend towards a very small number of suppliers or a monopoly. It will only be possible to achieve competition in such a situation by taking political decisions that assist new market participants in entering the market.

Chapter 5: Economic analyses of the use of open source on the desktop.
The economic analysis of the application of open source to desktop software is based on the costs of office suites. The analysis is made as a comparison between Microsoft Office and StarOffice/OpenOffice, and the software is assessed from both the rights and development perspectives. In switching from Microsoft to StarOffice/OpenOffice direct savings can be made in licence payments and costs of replacing software (as StarOffice/OpenOffice makes different/lower demands on hardware). On the other hand, there may be increased costs in building up expertise among the systems managers and in training end users. There are also problems with compatibility, particularly with layout and complicated spreadsheets.

Chapter 6: Economic analyses of open source as infrastructure software.
The economic analysis of open-source software as infrastructure software is based on foreign studies and a case study. The foreign studies show that open-source software is cheaper than proprietary software for the selected areas of application. The same is true for the case study. The analysis shows that open source as infrastructure software entails substantially lower costs.

Chapter 7: Analysis of open source and custom built software.
e-government will necessitate major investment in custom built software over the next few years. Ensuring sufficient competition afterwards, when the system has been developed, will be a problem for public purchasers. Ownership of the source code is essential for the later changes and adaptations, partly because the systems have to be put out to tender under the EU Directive. Many institutions need custom built systems, but do not want to be tied to a particular supplier. Proprietary systems entail a strong tie to a single supplier, and in reality this precludes competition, so that the EU's rules on tendering do not have any practical impact. User-owned systems are more expensive in actual development, but provide an opportunity for greater competition in continued development, and are therefore cheaper in the long run. An alternative for systems with several users is to develop the systems as open source and consequently bring about greater competition in the development of systems.

Chapter 8. The socio-economic consequences of open-source software.
There is significant socio-economic potential in the application of open-source software. The report asks to what extent open source is a genuine alternative in e-government. Economic estimates show that there is great economic scope for investments in both IT skills and pilot and development projects in choosing open source as an alternative to proprietary software under the prevailing economic market conditions in a number of software areas. Whatever choice is made, it will be necessary for decision-makers in the public sector to develop strategies for future IT investments involving open-source software.

Recommendations of the working group

The working group recommends that the government and other authorities should jointly formulate principles and objectives for the procurement of software, partly on the basis of the following observations:

It is necessary for a number of decisions in relation to IT to be taken in a coordinated manner, where the government – with all the ministries and agencies, etc. – is capable of acting as a corporation and taking joint decisions on the basis of a multi-year planning horizon.

Joint decisions are necessary to introduce open standards, which is essential if greater competition is to be established, with the application of open source as one of the options. Central decisions are also necessary to provide economic support for pilot projects and to draw up framework agreements, draft contracts, etc., that can serve as tenders or alternatives for local decision-makers.

In the short term

- The state must not put all its eggs in one basket. It is important to ensure for all types of software that each individual administrative unit has a real choice in a competitive market.
- Open-source software must be judged on the same terms as proprietary software, and in calls for tender and other purchasing open source must be assessed on the basis of a realistic costing which takes account of all economic factors.
- Investment decisions can often represent a mixture of open source and proprietary software. It is not an either-or decision, and the purchase of open source should not therefore be dictated as a general principle.
- An initial pilot project must be established in the near future in which open-source software such as StarOffice/OpenOffice is implemented in medium-sized e-government. The pilot project will be used to gather experience of the overall user-friendliness and quality of the systems, of the accomplishment of the change-over task, for example the training of users and IT staff, and the extent and resolution of compatibility problems in connection with electronic exchange in Microsoft formats. This experience must be put at the disposal of all other administrations.

In assessing options, special priority must be given to the value of the open source code, including the long-term value inherent in supplier independence with respect to maintenance and, the possibility of security being subjected to independent reviews.

In the longer term

- Establishment, for example within one to one and a half years, of a larger follow-up project in which a number of administrative units use open-source software, for example switching over to StarOffice/OpenOffice, and utilise previously gathered experience to reduce installation and adaptation costs.
- Preparation of a strategy for the introduction of an open standard for the exchange of word-processed documents.

The working group recommends that a standard document format be developed, firstly for problem-free exchange of documents and secondly for integration in systems used in e-government. A strategy for the introduction of an open standard for the exchange of word-processed documents is important, because there is no genuine competition at present in the desktop area, largely due to the fact that Microsoft formats also represent *de facto* standards for electronic document exchange, and among these the doc format for word processing is the most important.

Introduction

Open-source software (software with open source code) has become very popular in the last few years and is advancing at a speed unknown outside the world of IT. Just a few years ago, open-source software was regarded as a slightly 'nerdy' rebellion against the giants of IT. Today it appears in television advertising in the company of the IT giants. In other words, open-source software has become an area of business – an alternative, and therefore a competitor, to proprietary software.

This interest has also spread to the world of politics. Because open-source software is now increasingly used for commercial purposes, because it is characterised by independence from software producers, because it is opposed to the creation of monopolies and because it is characterised by a 'free-of-charge principle', open-source software has had a great impact on the political agenda, nationally and internationally.

This report does not judge whether open-source software is better in general than proprietary software, or whether it is more morally sound. It analyses whether open-source software is, or may become, a genuine alternative to proprietary software in e-government. The report also aims to take the discussion forward into the e-government of the future and to look at open-source software as a means of political strategy in developing this.

The working group has attempted to make a serious assessment of the advantages and drawbacks of using open-source software, without excluding any of the options at the outset and without looking at open-source software as an either-or problem. At the same time, the report identifies opportunities for putting open software to use in e-government. Open-source software opens new doors, and the report shows that the first step can be taken.

The report analyses open-source software against the background of the working group's internal discussions, an independently prepared economic analysis, various background reports, Danish case studies, the holding of a workshop and a long series of discussions with IT players.

Open-source software exists in many versions. This report focuses on the prospects for applying open-source software in public administration. The working group has therefore concentrated on the needs which exist within typical administrative units. This in turn has meant that the focus in the report is on office software (desktop) and operating systems (infrastructure). But as the working group has carried the discussion on open-source software forward into the e-government of the future, custom built software has also been looked at.

The working group does not consider the report to be exhaustive in either discussion, analysis, field of study or conclusions. It has, on the other hand, been the working group's endeavour to make a balanced assessment of the opportunities for using open-source software in public administration and to contribute an objective economic analysis of the social consequences of changing over to open-source software.

Chapter 1

Open-source software in e-government

This chapter introduces open-source software in connection with the development of Danish administration towards what is referred to as 'e-government'.

Open source designates software which is developed and maintained according to principles of far-reaching user rights, which include the right to alter, copy and distribute the software. This software policy is of interest today, because development of IT over the last ten years has been dominated by the Internet, and the Internet is fundamentally made up of open-source technologies. The expansion of the Internet has made this network the hub for the information exchange of both enterprises and authorities and increasingly also in important work processes.

Open source today comprises more products than those linked closely to the Internet. There are open-source software, operating systems, cooperation systems and special systems. At the same time, many of these systems benefit from the Internet, which makes possible digital cooperation between geographically separated units.

The public sector in Denmark and the whole of the EU faces a change-over to digital communication in the supply of public services, with the citizen at the centre. This means that coherent services have to be delivered to a greater extent, necessitating increased cooperation between administrative units and between the levels of the administration of the EU, national government, county and local authority.

This development makes great demands on the IT systems on which e-government is based, for instance with regard to options for exchange between authorities and with citizens and on the security with which this exchange takes place. This chapter outlines the challenges and objectives in e-government, and explains why the use of open-source software may be of interest to the public sector.

1.1. Open-source software in e-government

A change-over to e-government will necessitate huge investments in IT over the next few years. It is therefore natural that a close assessment is made in connection with these investments of the forms of information technology it is intended will be applied, and who controls the development and ownership of the fundamental technologies in e-government. This increases interest in the opportunities opened up by open-source software, and makes the question of the potential for the use of open-source software in e-government economically advantageous and relevant.

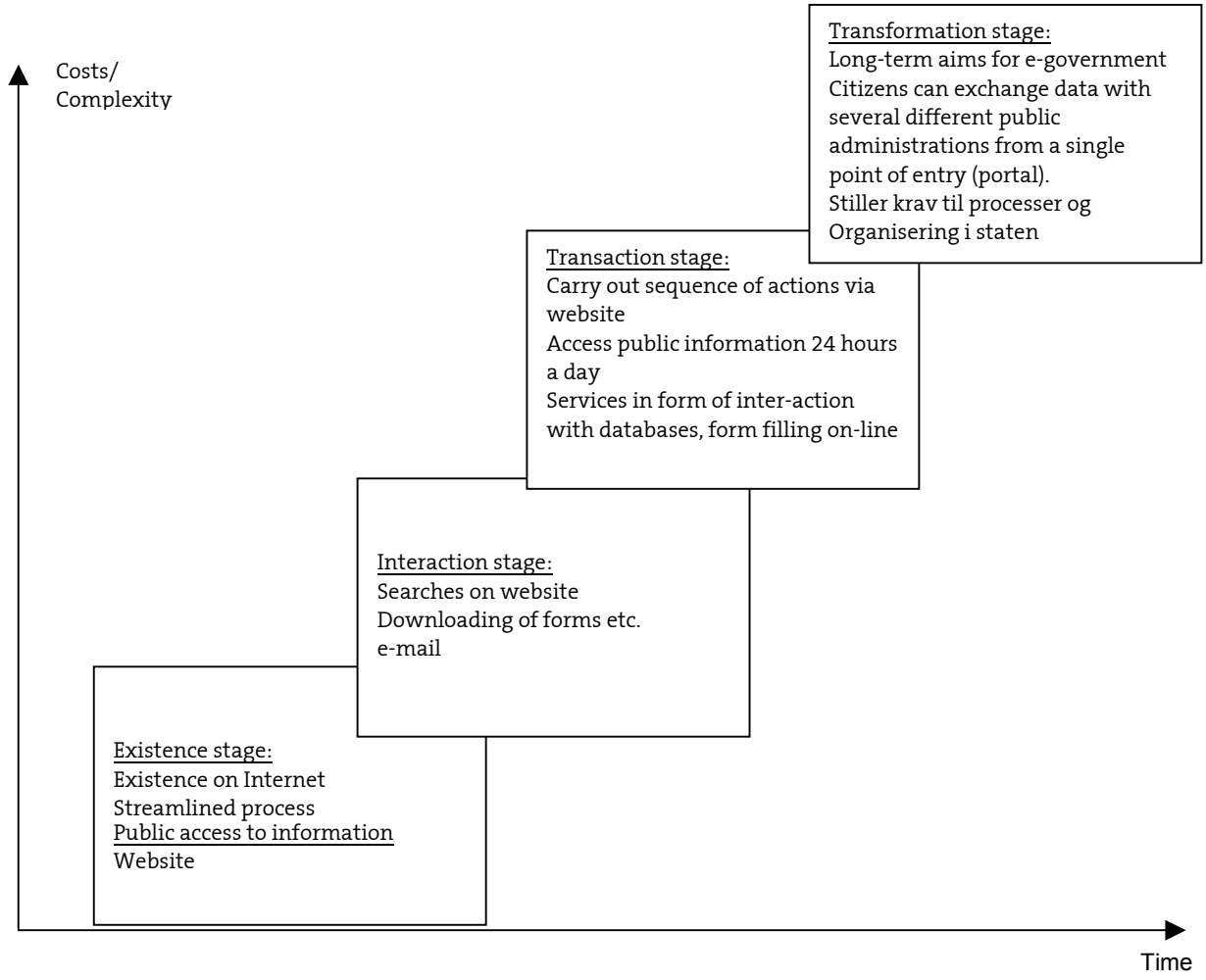
1.1.2. e-government and the requirements to be met by the public sector

The development of information technology over the last decade has left clear marks on society. In particular, the emergence and expansion of the Internet has revolutionised the way we communicate and affected organisation and interactions in society. The public sector has also 'gone on-line' – at first with general information on institutions and the work of authorities. But gradually, as the public and companies have become more accustomed to making bank transactions and shopping through the Internet, it has been increasingly expected that the public sector is also capable of supplying more complex public services. As a result of technological development in the public sector in general, it is said that 'e-government' is being developed. This can be described in practical terms as follows: *'Use by public administrations of technology, primarily Internet-based technology, which contributes to improving access to and supply of public information and public services to citizens, companies and the public sector as a whole'*.[1]

It is therefore expected that e-government will have the potential to create easier and smoother interaction between the public sector and the citizens, but are still at the beginning of this development. A model presenting a schematic overview of development in e-government is shown below:

[1] Developing fully functional E-government: Four Phases of E-government, Gartner, 2000.

Figure 1.1. Phases of development in e-government

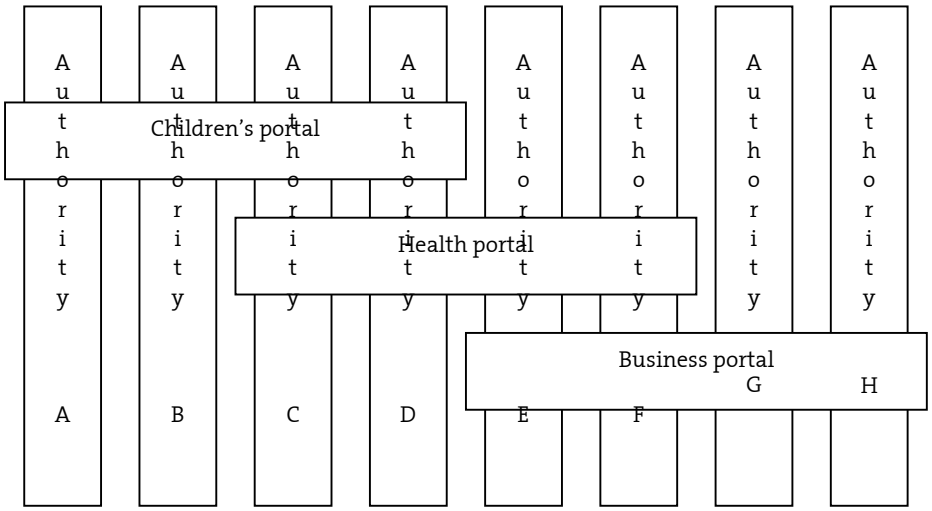


The vast majority of public institutions now have their own websites, with general information on the aims and work of the institution, which relates to the first phase in the model: The *existence stage.* Many public institutions have gone a step further to the *interaction stage* and today additionally offer more advanced websites, where the content is adapted to a greater extent to the individual user, and where users are able to interact with the websites to some degree. The users (citizens) have the option, for example, of downloading forms, information, guides, can carry out information searches, communicate with institutions by e-mail, and so on. Some public authorities have reached the *transaction stage* and offer websites where actual transactions can be made. This may involve students, who have the option of enrolling for university courses on-line, or citizens submitting their tax returns through a tax authority website.

Public administrations for centuries have been made up of vertical units, which supplied a particular product or service to society. If a particular task was to be accomplished as a citizen or enterprise, it was necessary to apply to the series of authorities dealing with the task concerned.

As a result of technological development and citizens' increasing expectations for the level of public service, there are now calls for government service systems to be put together on the basis of public's need for a readily accessible and logically coherent chain of transactions. It has to be possible for services to be provided through one point of entry, an Internet portal, which integrates all the services which members of the public or enterprises might need in a particular situation. It may be a portal aimed at families with children, which offers all the services a young family requires, regardless of which public body is to supply the service. Figure 1.2 illustrates how Internet-based services (portals) have to work across authorities.

Figure 1.2. Internet-based services (portals)



Portals of this kind call for different connections in the procedures of the various public institutions, and only a few public services are supplied in this way at present. The *transformation stage* requires significant changes in the public sector, not just in the construction of IT system, but above all in procedures. e-government is therefore still some way from fully utilising the potential that exists in the use of information technology and the Internet in particular.

1.2. e-government in Denmark

Development is, however, well under way, and e-government comes high on the political agenda, including in Denmark. This was recently emphasised by the two party leaders, the Prime Minister, Anders Fogh Rasmussen (Liberal Party), and the Minister for Economic and Business Affairs, Bendt Bendtsen (Conservative People's Party)[2]: *'The Government wishes to make it easier for members of the public to deal with the authorities. This will be one of the areas focused on by the Government in the future [...] ... 'Members of the public must be able to contact public authorities 24 hours a day through the Internet. Businesses must be able to obtain all forms and information from one place [...]. Young families must also be able to find information on child allowances and childcare facilities in one place,[...]. Members of the public must be able to see their own data in public registers and be able to track their cases in the public sector on the Internet. This creates greater openness and more democracy.'* In May 2002, the Government published its modernisation programme, which states: *'The Government wishes to put Denmark at the forefront as a modern IT and knowledge-based society and apply new technology to alter the way work is done in the public sector. The aim is to provide the public with a better service and release public servants for tasks which make a direct contribution to public welfare. Efficiency improvements and less bureaucracy are achieved when the entire accomplishment of the task is thought through. The Government therefore wishes to explore new avenues in all areas and challenge established procedures and paper-based routines... A commitment must be made over the next few years to digital solutions that do away with manual routine case processing and reduce administration. The outlook is for more efficient public procedures and faster and error-free processing of cases, to the benefit of the public and businesses....New technology must help create increased cooperation across the public sector and traditional sector boundaries. Taking account of the legal rights of members of the public, it must be ensured that information can be exchanged between public IT systems, so that members of the public experience the public sector as being an efficient entity which works smoothly. In that way, duplicated work is avoided and members of the public do not need to provide the same information repeatedly.*

1.2.1. From centralism to decentralism and back again

From the historical point of view, development in IT in national government has gone from centralisation to decentralisation and is now on the way back again. The administration of IT purchases by national government was originally the preserve of the Ministry of Finance, which had to approve all purchases[3]. As a result of a rising number of purchases and investments in IT, it was not possible to approve all purchases, and in 1992 the approval procedures were therefore abandoned for national

[2] Statements made following the government seminar in April 2002.

[3] Circular of 15 October 1964 stipulated that all investments in automatic data processing were to be approved by the Administration Department of the Ministry of Finance.

government IT purchases, and IT investments by national government were in reality made freely.

The report 'Info 2000' from 1994 recommended that national government should start discussing how IT could be developed in the state sector, and provided the basis for development of an actual IT strategy. In 1994, major tasks relating to IT in national government were transferred to the Ministry of Research from the Ministry of Finance, resulting in the preparation of annual action plans for IT development and application.

In the autumn of 2000, the Ministry of Finance appointed the 'E-government Committee', an interdisciplinary committee charged with ensuring the development of e-government across existing organisational boundaries, ensuring better and cheaper public service and improving the efficiency of procedures by introducing e-government. The Committee was composed of representatives from the National Association of Local Authorities in Denmark, the Association of County Councils, Copenhagen Local Authority, Frederiksberg Local Authority, the Ministry of IT and Research, the Ministry of Business Affairs, the Ministry of Internal Affairs and the Ministry of Finance, which also provided the chairman. Together they made up the Digital Task Force.

1.3 e-government according to the Digital Task Force

The Committee published a report on e-government in 2002[4]. The vision for digitised public administration was defined in this report as: *'improved and more efficient accomplishment of administrative tasks through the application of information technology to the benefit of members of the public, businesses and the public sector.'*[5]

In lending weight to IT-supported development of procedures and organisation, e-government differs from IT application in the public sector to date, which has primarily entailed technical support of existing procedures.

It is important to point out that e-government, which both definitions establish, is concerned with improving the level of service to society using technology. The report from the Digital Task Force has identified four pointers which should guide work on e-government. The four targets are:

Target 1: *e-government should actively contribute to the development of network society*: The public sector can do this by virtue of its size, for instance by being involved in the setting of standards for IT use and increasing IT skills broadly among the population.

Target 2: *The public sector should work and communicate digitally:* This involves digitising work processes in relation to members of the public and companies.

Target 3: *The services of the public sector should be delivered in a coherent way with citizens and companies at the centre:* It should be possible for citizens to communicate with the public sector by accessing one place. It should be possible to access this place 24 hours a day, 365 days a year.

Target 4: *The tasks of the public sector should be carried out where they are best dealt with:* e-government should break down administrative boundaries and ensure that specific work tasks are handled flexibly in a more dynamic organisation. Sharing of knowledge between the different parts of the public sector should be expanded using information technology. As work processes are digitised, not only will processes be changed, there will also be a change in the organisation in order to raise the level of service and efficiency.

1.4 e-government in practice

A large number of projects of varying size are already under way in the public sector to promote the development of e-government. Some examples are:

[4] Projekt Digital Forvaltning (2002) 'På vej mod digital forvaltning – vision og strategi for den offentlige sektor', www.e.gov.dk.
[5] Report on e-government, Ministry of Finance.

Project	*Contents*
Digital Signature	The aim of the project, in conjunction with private players, is to establish the necessary and sufficient technical and organisational frameworks for the introduction and use of digital signature on a broad basis in society.
The Family at the Centre	This project is to analyse the present situation and propose coherent digital solutions for services and situations, where young families (children 0-6 years) are in contact with the public authority, for example in connection with childbirth, day nurseries, maternity leave, childcare, child benefit/allowance and nursery school, and contact between home and institutions.
E-Citizen	In cooperation with the National Association of Local Authorities in Denmark and the Danish State Information Service, this project is to assess the need and opportunities to create one or more attractive interdisciplinary common citizen portals. These portals will contribute to better citizen service and improved efficiency in the public sector.
A collective integration process	Increased cooperation with smooth exchange of information between local authorities and the central immigrant integration authorities will improve both quality and the efficiency in the individual immigrant's contact with the public sector.
From Hospital to Home Care	This project will improve electronic communication between the local authorities and hospitals in connection with admission and discharge of patients, so that a more holistically oriented effort can be made.
Industrial injury cases	Contact takes place with many parties in the processing of industrial injury cases. There is great potential in digitising this interaction. There is much to be gained in particular in the interaction with the municipalities by re-using medical assessments which today are requested by both local authorities and the National Board of Industrial Injuries. The aim is for the citizen to have to supply information only once.
One account for enterprises in the public sector	This project is intended to prepare for a decision on collective account for public-law payments by enterprises and financial holdings with the whole of the public sector. This will mean large efficiency gains for both the public sector and enterprises – particularly if terms of payment, interest rates etc. are harmonised at the same time.

1.5. Information Technology requirements of e-government

Such extensive dependence on information technology makes it necessary to consider carefully which technologies are selected for use in e-government. What are the total expenses, both direct and indirect, in using a particular technology? How does a given technology work together with other technologies that are used in connection with e-government? What opportunities and rights does the user of the technology have? How well does a technology work? How is data exchanged most smoothly between the systems? What are the social consequences of a given choice of technology?

Security is the key

Security is one of the key topics in relation to e-government. Increased use of technology means that many items of information will be stored electronically, and it is therefore necessary to rely on the software used not containing elements that permit external access or opportunities for undesirable release of information to third parties.

Finance and efficiency

The aim of e-government is to improve the service provided to society, for example in the form of more efficient electronic processing of cases. Improvements in service via the Internet at the same time mean financial savings for the public sector in the longer term. Some key topics in relation to finance and the debate on efficiency are presented below.

Work processes

The aim in introducing full e-government, as mentioned, is not just to add a technological solution to existing public administrative routines and procedures. Efficiency gains in introducing e-government are dependent on

work processes being re-assessed and modified. Information technology therefore has to support adjustment and simplification of existing work routines. Against the background of a survey of 65 government institutions, the Ministry of Finance has calculated that there is efficiency improvement potential of between 17 and 31 per cent in administrative resources in changing over to electronic administration. This is equivalent to between 2 and 17 per cent of total operation, depending on the type of institution concerned.[6]

Integration of systems
But the potential for efficiency improvements depends just as much on the extent to which the internal administration in government is integrated with the systems that are in contact with the world around and this integration proceeding across existing organisational boundaries. If e-government is to be effective, it is necessary for the systems to be able to communicate with one another. There is a need to standardise such solutions so that it is possible for systems to be integrated and data re-used. As we shall see later, open standards are essential for such development to succeed. It is therefore appropriate to focus on how public data is exchanged and stored.

Management
Several public institutions will shortly need to develop new IT systems. A key question in relation to both finance and efficiency is how these systems are to be developed. Broadly, two questions can be asked on the management of development: is it to be done using central management, which may mean that the system is not adapted to local needs, or is the development to take place in a decentralised way, which may mean that the same work is performed and paid for several times?

National government as purchaser
A close relationship may arise between supplier and buyer in the purchasing of software. This often leads to dependence on the supplier, often referred to as a lock-in effect. Once a system has been acquired, the user is dependent on when and how the supplier modifies the product. A number of supply rules have been developed to counteract lock-in, but the question of whether these are adequate remains. Consideration therefore has to be given to how national government, as the purchaser, is to act in relation to possible suppliers of systems, including questions on updates and enhancement of the software.

National government as a financial player
The size of the public sector in western European societies means that public provisions are of great significance to the private business community. One area in which this applies is procurement of technology. In making purchases, authorities are subject to a requirement not to favour individual suppliers unfairly. At the same time, the size of public purchases, if they are coordinated or merely coincide, means that selected products may gain market-leading status in the Danish market depending on the extent to which the product contains imports or exports. The investments necessitated by the concept of e-government could therefore have effects on market structures and the development of prices on Danish markets, while they will not significantly affect global market products.

[6] Ministry of Finance publication Digitalisering og effektivisering i Staten – May 2002, p. 50.

1.6. Why is open-source software of interest to the public sector?

To date, the public sector has mainly made use of proprietary software. One of the many technology choices the public sector now faces is the question of whether open-source software has sufficient functionality and user-friendliness in comparison with proprietary software and whether it is cost-effective from an overall perspective.

Open-source software offers a number of possible advantages over proprietary software. These advantages will be outlined below, while a definition and a discussion of open-source software are provided in Chapters 2 and 3. The economic aspects of open-source software are discussed in Chapters 4 to 7 and summarised in Chapter 8.

Possibility of savings
Open-source software is not necessarily free, but in most cases is considerably cheaper than proprietary software. There may therefore be economic advantages for the public sector in using open-source software.

1.6.2. Open standards and possibility of integration

The characteristic feature of open-source software is that it uses open standards. The expression open standards means that the principle for the development of software is established in public forums, in contrast to proprietary (industry) standards, which are kept secret. It also means that it is often possible, through a more or less democratic mechanism, to influence the standard. With e-government, it is necessary to be able to read and exchange data without encountering problems. It is therefore appropriate to focus on how public data is exchanged and stored.

Possibility of access
Many items of information in e-government will be stored electronically. It is therefore necessary to rely on systems not containing elements which permit external checking, or undesirable release of information to third parties. Systems with freely available source enable both government and other experts and members of the public to carry out post-inspection of the programs used. It is therefore not possible, for example, to hide what are known as 'back-doors', i.e. secret entrances, in an open-source software product, as these will be discovered when the code is inspected.

Security and quality
Open source code enables the user and other interested parties to check whether the program is written in a justifiable way and where appropriate to identify elements of danger for

the stability and security of the code. This makes it possible for open-source software to be secure and of high quality.

Supplier independence
A 'lock-in' situation can easily arise when software is purchased: once one system has been acquired, one is tied to it and becomes dependent on when and how the supplier modifies the product. This 'lock-in' will not arise if open-source software, which is based on open standards, is used. Data is not stored in a proprietary format, and it is possible for users to change between several different systems and therefore also several different suppliers.

Possibility of customisation and re-use
As the source code is open and it is permissible to modify this, public authorities can adapt open-source software to particular needs. It will be possible for a customisation to be re-used in other parts of the public sector, resulting in lower software customisation costs.

1.7 Open-source software from the political point of view

Open-source software has attracted attention in the political system. In 2000, the Research Committee of the Folketing, the Danish Parliament, adopted a report on 'Proposals for a Folketing resolution on a strategy for the expansion of open-source software in Denmark'.[7] In this report, the Committee recommended that the Government should contribute through national IT policy towards providing information on the potential offered by open-source software and should recommend that the use of open-source software be included as an option in invitations to tender relating to IT. Several European countries have launched initiatives on the use of this type of software. This has meant either replacing existing systems with open-source software or making recommendations on future use of open-source software.

Germany
The German central administration in June 2002 entered into a framework agreement with IBM and the firm of SuSe on the supply of open-source products, based on Linux. The agreement makes it possible for German public administration to acquire Linux-based systems at a reduced price from IBM. The agreement makes it possible to supply servers, as well as Linux-based workstations. IBM will provide ongoing support for use of the systems. The German Government wishes to promote alternatives to Microsoft with the contract, but this is not a law, more an offer to the public decision-makers.

UK
The British Office of E-envoy issued a British policy in the area of open source at the end of July 2002. It is stated in the open-source policy that the British Government and British authorities will in future consider open-source software systems on an equal footing with proprietary solutions when purchasing IT. It is also an objective that the British Government will in future as far as possible use products based on open standards. In principle, the British Government wants to get 'as much as possible for its money' in IT purchases, which is one of the arguments for considering open-source software. In addition, the UK Government wants to avoid problems in the future with 'lock-ins' in relation to a specific supplier.

France
The French Government has decided that French central administration should terminate its agreement with Microsoft on the supply and use of their software. This decision means that all French national and local authorities as far as possible are to use open-source software. This is software where the program kernel is to be publicly available, and where the finished programs are freely available and at no charge, for instance via the Internet.

An office, the Agency for Information and Communication Technologies in Administration (ATICA) was set up in August 2001 to follow up the French Government's decision, to coordinate the IT initiatives in and between the public authorities. The Agency is to ensure that public IT projects make use of open source standards with a view to ensuring interoperability and reducing expenditure on IT.

Finally the French Government wishes to improve the opportunities for small companies in the software area, by making it possible for them to work on public open-source projects. The aim is to support the enhancement of open-source software.

Other countries
Several open-source software initiatives have seen the light of day both in Europe and in the rest of the world. In Finland, some members of parliament led by Kyösti Karjula have issued a recommendation on the use of Linux platforms in public administration. A working group appointed by the European Commission has issued a report that recommends public administrations in the Member States to use and swap experience relating to open-source software. Finally there has been some discussion in Peru on a legislative proposal compelling public authorities only to buy and use open-source products. The arguments in favour of the use of open-source software here are based on the possibility of access, the permanency of data and the security aspect.

1.8. Conclusions

The public sector is to be changed over to communicating electronically and, by using the Internet in particular, is to provide public services with the citizen at the centre. This means coherent services are to be provided to a greater extent, across administration boundaries.

The change to e-government also means great challenges economically, as huge investments will have to be made in IT in the public sector over the next few years. It is therefore appropriate in connection with these investments to assess closely which forms of IT technology it is intended will be used and what

[7] Report issued by the Research Committee on 2 October 2000, draft resolution B114.

effects on market structure the arrangements made by government can be expected to have. In this context, it is essential to view open-source software as a serious alternative to proprietary software.

Chapter 2

What is open-source software?

Who would buy a tin of tomatoes with no product label on it, or who would buy a car with the bonnet welded shut? These are the kind of rhetorical questions advocates of open-source software ask to emphasise that the purchaser of proprietary software has too few rights. When software is open source, the source text is available, the user has the right to modify it and several users can exchange improved versions among themselves. In traditional software, which we refer to in this report as 'proprietary software', users are unable to modify the software themselves. When there is a need for bugs to be fixed, for example serious security defects, users are dependent on the ability and willingness of an individual supplier to supply these -like the unfortunate owner of a car, where the engine can only be repaired in the supplier's workshop and on the supplier's terms.

Open-source software is thus linked to the question of rights to software. This chapter introduces open-source software by first outlining how it relates to a practice determined by attitude and movement among developers and users of software. The remainder of the chapter describes rights as laid down in licences for open-source software. The licence for the open source web server Apache is examined by way of example. There then follows a general overview of the rights the user has to open-source software. Acquisition of open-source software is generally – but not always – free of charge, and this is looked at in depth, after which there is a discussion of the relationship between open source and open standards. The statutory basis for software licensing consists of copyright legislation, which is finally outlined, together with an explanation of the working group's choice of the term 'proprietary'.

2.1. Open source as practice and movement

The Free Software Foundation is the best known of the early organisations that worked to promote what is now referred to as open source. The Free Software Foundation (FSF) was set up in 1985 by Richard Stallman to support the development of a range of software including operating system, compiler and editor. The aim was to create such an extensive software package that users had a complete productive desktop at their disposal, without any proprietary software.

The Open Source Initiative (OSI) was formed in 1998 to establish a pragmatic alternative to FSF. The people behind OSI, including Eric Raymond, regarded FSF as ideological and confrontational, and wanted to establish a platform to disseminate the open source ideas in a more pragmatic way, particularly towards the IT industry.

2.1.2. Sharing of software and knowledge

A common attitude in the open source community is that the restrictions on the rights of users of proprietary software prevent the sharing of software and knowledge, and that this conflicts with the user's interest but also in a broader sense with the objective of promoting the development of software technology.
Another important element in attitudes in the open source world is support for open standards. The HTML standard is an example of an open standard. A particular characteristic of open standards is that their definitions are publicly available in contrast, for example, to Microsoft's doc file format, which is secret. Open standards promote user independence. For example, anyone can attempt, on the basis of the definition of the HTML standard, to develop programs that write or read HTML documents (e.g. web browsers).

2.1.3. The open philosophy of the Internet

The development of the Internet from its beginning in the 1980s and 1990s was in several ways linked to the open-source philosophy. The central network protocols etc. were developed in an open standardisation process under the auspices of the World Wide Web Consortium and the Internet Engineering Taskforce, partly on the basis of experience gathered from reference implementations of the standards. These were programs in which the source text was available and were distributed under open-source-like licences.

Part of the development work had roots in universities and their tradition of openness and publishing results. The Internet, with electronic mail and newsgroups, was the communication channel for the groups who in the nineties developed open-source programs such as Apache and the operating system Linux. The successful development of these programs showed that extremely complex programs could be developed as open source. A generation of software developers, network people, system administrators etc. acquired positive experience of the spread of technology based on open standards and available source text by participating in the development of the Internet.

2.1.4. Patents

Advocates of open source have entered the debate on the introduction of patents on software, which is regarded as a disincentive to the development and spread of new knowledge in the area of software. Software patents can create difficulties for open-source projects, as these projects rarely have financial funds at their disposal to pay for patented methods. Software patents are not discussed in more detail in this report.

2.1.5. Open source can be commercialised

Is open source compatible with commercial business strategies – or does the concept stand and fall with idealism and volunteer labour? As an open-source licence entitles the user to copy and distribute the software, the supplier's ability to sell licences is undermined. By far the greater part of the costs of software is accounted for not by the original development of the software, however, but by services in the form of customisation and maintenance, and the open source model is not an obstacle to firms selling such services.

Many private companies are involved in the development of open-source software. Sun and Netscape have implemented large open-source development projects (the office package OpenOffice.org and the browser Mozilla respectively). IBM contributes to the development of the open-source web server Apache and the operating system Linux, and in 2000 announced that the firm planned to spend 1 billion US dollars on developing open-source software.[8]

2.1.6. How far has open source come?

The development of open-source software has reached the stage that today it is possible to create a fully functional workstation that is based solely on open-source software, and that can be used as a tool in advanced software development – or for word processing and other functions used by ordinary users in the public sector. It is also possible to build up all the functions included in a modern website. It must be emphasised that this report is not an analysis of whether the public sector should completely change over to open-source software. The purpose is to analyse how the public sector can make the best possible use of the potential in open-source software with different forms of licence, and in each case choose the software that provides the best solution for a particular task.

2.2. Example: The Apache web server and its licence

An example of open-source software is the web server Apache. Apache is the most widespread web server in the world. The British firm Netcraft regularly surveys the spread of different web servers in the market. For several years, Apache users have made up more than 50% of Internet users. The Netcraft survey in July 2002 covered over 37 million web servers, and shows that Apache accounted for 57% of these.[9]

The user's rights are laid down in the Apache licence, which runs to a total length of about one page, and the core sections of which read as follows:

'Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must [.. same conditions]
3. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: 'This product includes software developed by the Apache Software Foundation (http://www.apache.org/).'(..)
4. The names 'Apache' and 'Apache Software Foundation' must not be used to endorse or promote products derived from this software without prior written permission.(..)
5. Products derived from this software may not be called 'Apache' (..)
THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES (..) ARE DISCLAIMED. (..)
This software consists of voluntary contributions made by many individuals on behalf of the Apache Software Foundation. For more information on the Apache Software Foundation, please see <http://www.apache.org/>.
Portions of this software are based upon public domain software originally written at the National Center for Supercomputing Applications, University of Illinois, Urbana-Champaign.[10]

The licence thus gives the user the right to use, modify and distribute. A user who sells or in some other way distributes software containing Apache is obliged to draw attention to the fact that Apache is included. The licence also contains an exclusion of liability and an acknowledgement that the roots of the Apache software go back to the University of Illinois.

IBM's Websphere product family for the development of websites among other things with electronic commerce contains Apache as one of its key components. Under the Apache licence, IBM has to mention in the product material and similar documentation that Apache forms part of the product, and IBM is otherwise free to use the Apache software. IBM has made a major contribution to the development of Apache, and several members of the Apache Software Foundation group, which directs the Apache project, are employed by IBM.

2.3. The user's rights according to open-source licences

Open-source licences give the user:

- access to the source text
- the right to use
- the right to modify
- the right to distribute

In the view of the working group, this definition captures the essence of what is generally understood by open-source licences. The definition is consistent with, but less extensive than, the one

[8] New York Times, 20 March 2002, cited in J. Feller and B. Fitzgerald: Understanding Open Source Software Development, Addison-Wesley, 2002, p. 3
[9] See http://www.netcraft.com/survey
[10] (Source: http://www.apache.org/LICENSE, dated 31 July 2002)

given by the Open Source Initiative (OSI).[11] The organisation runs a certification service on the basis of its definition, and has approved the Apache licence and more than 40 others. One of the functions of OSI is to shed light on those licences that have been used when firms have released software already developed as open source, and it has sought to establish open-source projects led by the firms themselves. In this situation, OSI's assessment of the licence may be of interest to potential participants outside the firms, who have wanted to make sure that the often complex licences were 'proper' open-source licences.

The Free Software Foundation early on developed what is known as the GNU licence, which is more far-reaching than the minimum requirements in the OSI definition. The GNU licence requires that modified versions of the original software are also made available to other users under open-source licence terms. In contrast, licences such as the Apache licence allow for a firm to 'close' an improved version of Apache and try to sell traditional licences. These differences are substantial for firms that work commercially with open source, but not for government as a user of software products, and are not considered in more detail in this report.

Access to the source text means that the software can be 'reviewed' by experts who are independent of the supplier. When software is only supplied in binary, machine-executable form, on the other hand, it is not possible in practice to recreate the source text, and it may also be prohibited under the software licence. Information on the features of the software can therefore for the most part be obtained only by studying how it behaves during execution, which gives far less of an insight than if one also has access to the source text.

The right of use means that no obstacles are put in the way of using the software for example for commercial, military or other purposes. The right of use also means that open-source programs may be used together with traditional, proprietary software. A public organisation is therefore at liberty to choose a 'hybrid' strategy, where both open-source and proprietary software are used.

The right to modify gives the user organisation the right to carry out any form of customisation, debugging etc. of the software it might want. The right to modify is linked to access to the source text, without which it would not be possible in practice to make a desired modification. The right to modify can be utilised internally by IT staff, but also makes it possible for suppliers to be selected on the basis of such service tasks. The right to modify consequently provides supplier independence in relation to maintenance tasks.

The right to distribute means that the user is permitted to create several copies of the software and to distribute them, both internally in the organisation and to other organisations.

2.4. Distribution and payment

A consequence of the right to distribute is that open-source software can be acquired at distribution costs, which means either free of charge or for a small charge, for example for dispatch of a CD. Open-source licence rules do not, however, contain any provisions relating to the software having to be free of charge. The Apache licence, for example, does not contain any prohibition on accepting money for the Apache software, or for products containing Apache.

The question of payment for open-source software is covered by the first item in the OSI definition, which requires that open-source licences do not prevent or impose such payment:
'The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale'.[12]

When open-source software typically is free of charge, this is consequently merely a mechanism based on the right to distribute. This mechanism presupposes that the software exists beforehand, and is distributed to a customer who is willing to distribute it further. If the software does not exist in advance, on the other hand, and the customer therefore places an order with a supplier to develop it, the customer obviously has to pay for the development work, whatever the form of open-source licence.

On the basis of the compatibility of open-source licence rules with accepting money for the distribution of open-source software, there are firms that have this as part of their business concept. The firm Red Hat, for example, charges money to send a pack containing a number of CDs with the open-source operating system Linux and associated extra software.

Under the software licence, the purchaser of one of these packs can both freely copy the software within his organisation and make it available to others as copies on CD or via the Internet. The firm Red Hat and other distributors of open-source software will therefore only be able to induce users to pay a price equivalent to the convenience of having a ready-made CD instead of having to burn one, the certainty of knowing that the CD actually contains the particular software suite Red Hat has selected and other accompanying products, such as a book containing documentation on the software.

In addition, the purchaser can choose to pay a premium purely out of sympathy for the open-source idea and to assist in securing an economic basis for the continued existence of the distributor, but this hardly constitutes a major source of income.

2.5. Open standards

Open-source software preferentially uses open standards. Just as no principle of being free of charge is laid down in the open-source licences, neither is the use of open standards stipulated in them. The licences only regulate the relationship between the copyright-holder and the user, and do not describe in detail what type of standards or other methods are used in the software.

The standards concerned are definitions of file formats, layout, protocols etc. The standards typically define interfaces between the software and its surroundings, for instance standards for how the software transfers data to and from files on the computer, or in exchange with the programs of other computers via the Internet.

[11] http://www.opensource.org

[12] Source: http://www.opensource.org

An open standard is understood fundamentally as being a published definition. To take some examples, HTML and TCP are open standards, because they are published - by the World Wide Web Consortium (W3C) and the Internet Engineering Task Force (IETF) respectively, which have been responsible for developing the standards.

In a broader sense, the term open standard contains some other requirements, including that a standard is developed in a consensus process, which is not dominated by a particular supplier, and that the standard is not merely public in principle but is available free of charge or cheaply. It is supplemented by the standards developed in W3C and IETF. Unless otherwise stated, however, we use the term 'open standard' in this report in the narrower meaning of a published definition.

It is logical that open-source software preferentially uses open standards, firstly because open standards are in tune with the basis of values in open-source philosophy, and secondly because the use of a standard in an open-source program in itself signifies partial publication of the standard.

There is, however, no clear connection between the licence form of the software and the standards the software uses. Firstly, proprietary software can use open standards. Secondly, some source programs actually use closed formats. OpenOffice/StarOffice, for example, uses Microsoft's doc format. This is normally done as a matter of necessity, to ensure that the software can be used together with existing, closed formats, and in the case of StarOffice/Open Office the product has its own, preferred file format, which is open.

2.6. Licences and copyright

Copyright legislation is the basis on which the software supplier is able to use licences to stipulate rights (and obligations) for the user's application of the software.

Under copyright legislation, both in Denmark and in other countries, software is counted among works within the meaning of copyright law. Section 1 (3) of the Danish Copyright Act reads: *'Works in the form of computer programs shall be counted as literary works'*. It is not normally the case that the user signs the licence agreement. The normal situation is that the user implicitly accepts the licence terms by starting to use the software. An installation program often draws the attention of the user to the terms of licence – and to the fact that putting it into use is deemed to signify acceptance of the terms.

Copyright to proprietary software normally belongs to a commercial enterprise, and Microsoft, for example, holds the copyright to the Windows operating systems. In the case of open-source software, the copyright-holder may, for example, be the Apache Foundation, which holds the copyright to Apache, as a result of the developers of the software having assigned their copyright to it.

We have chosen to use the designation 'proprietary software' in this report for software with traditional licences. None of the designations known to us appear to be entirely pertinent, and the contrasting of proprietary and open source may be misleading, as the firm or the developers hold the copyright in both cases.

More specifically, we understand by proprietary software that the licence:

- does not provide access to the source text
- only provides very limited access to distribution, typically for purposes linked to the individual user's general use of the program, such as the creation of a backup copy

In the case of proprietary software, the licence may directly prohibit the user trying to recreate the source code on the basis of the binary form of the software, by decompiling or 'reverse engineering'. The Copyright Act lays down a limit on the extent of this form of prohibition, however, as far as interoperability is concerned. Section 37 stipulates that the user shall not be prevented from undertaking decompiling with a view to attaining interoperability. In other words, the user may try to recreate the source text for parts of the software, provided this is necessary to establish an interface between the program and other software.

Licences for proprietary software normally contain exclusions of liability fully equivalent to those contained in the Apache licence (Clause 2.2). A user who is 'hacked' and incurs large expenses as a consequence of a security loophole in the software cannot obtain compensation from the supplier.

There may be reason for the user of proprietary software to examine the more detailed provisions of the licence:

Microsoft licences for organisations with many users – known as volume licences – have a common part under the name of *Product Use Rights*.[13] The product use rights contain some controversial sections, which give Microsoft certain far-reaching rights in relation to the user:

- The rules entitle Microsoft to collect and store technical information about the user. The information may also be passed on to third parties, provided the user cannot be identified (p. 1-2)
- In the section on Internet-based service components, the licence stipulates that for some software components Microsoft has the right to automatically download upgrades and fixes to the user's machine (p.7)

Microsoft's gathering of information on the user can be done via the Internet, as the installation, activation or use of Microsoft's operating systems or Internet browsers may mean that the software automatically establishes a connection to Microsoft. Skilled users can, however, presumably protect themselves against at least some of the measures the licence rules entitle Microsoft to take, for example by deactivating particular features, or ultimately deleting certain programs.

Although proprietary software today is the traditional form of software, the severe restrictions on the user's rights are a relatively new phenomenon. In the 1940s and 1950s, software was predominantly developed in research environments. Commercial software development took place particularly at hardware producers such as IBM, who developed software to be supplied together with the hardware, and the customer did not pay separately for the software. Software to be supplied in return for payment was rare, and was

[13] http://www.microsoft.com/licensing/downloads/pur.pdf

normally special software, developed on the basis of contracts with a single large customer.

It was not until the late 1960s that the business model for proprietary software arose in the United States. An industry of software suppliers had developed there who were independent of the hardware producers, and who produced for a market where there could be several customers who were interested in the same software. It was in this connection that the supplier's strong interest in having sole right to distribute and therefore sell software licences arose, and adaptation of copyright legislation etc. to cover the area of software followed from this. In the early phase of the history of computers, on the other hand , it was general practice for the user to have rights to the software along the lines of those given by open-source licences, which are therefore in a sense the true 'traditional' rights to software.

2.7. Conclusions

The core of the open source concept is the user's access to the source text and right to modify and distribute the software, as laid down in an open-source licence. Open source is also linked to the development of the Internet and to attitudes on sharing of knowledge, freedom and open standards.

The right to distribute means that the open-source software – but not if it is specially developed – can be acquired at distribution cost, in other words free of charge or almost free of charge. Costs of adaptation, maintenance and so on, which make up the greater part of the total costs in connection with software, still have to be borne.

Open-source software gives the user a higher degree of supplier independence than proprietary software: the right to modify entitles the user to choose the supplier for maintenance tasks, and the use of open standards by the open-source software provides greater freedom of choice with regard to the other software it is to be used with.

Chapter 3

Open source as desktop, infrastructure and custom built software

In analysing the economic consequences of increased use of open-source software in government, the working group will divide software into three categories:

- software on the desktop (where the economic consequences are analysed in Chapter 5)
- infrastructure software (Chapter 6)
- custom built software (Chapter 7)

In this chapter, we present a definition of the three categories and emphasise some conditions for the introduction of open-source software specific to each category.

3.1. Definitions

Desktop software is the software that administrative staff use in their day-to-day work. We are thinking here in particular of office software such as word processing, spreadsheets and other programs in office suites, as well as web browsers, e-mail clients and calendar programs. This category comprises both software in the form of independent applications, for example for word processing, and software in the form of clients, such as e-mail clients, which require connection to a server.

Examples of infrastructure software are web servers, mail servers and operating systems. The infrastructure software usually constitutes the underlying element on which e-government and modern network society as a whole are based. It is an electronic counterpart to the transport infrastructure in the form of motorways and railway lines. The most important part of the infrastructure is the Internet and the World Wide Web, which among other things connect citizens with administration.

We understand custom built software to mean software that is developed for government with a particular application in mind.

3.2. Software on the workstation

Desktop software in the public sector is dominated by Microsoft products, including the Office suite with the word processing program Word. The dominance of Microsoft products means that Microsoft's formats represent the *de facto* standard for the exchange of word-processed documents in government. The most important of these formats is the doc format, used by Word. The doc format is referred to below as a file format. The format is used to store a document as a file.

In the short term, if open-source alternatives are to be more widely used, they need to be able to handle Microsoft formats. It is difficult to achieve this fully, because Microsoft formats are secret. There is therefore no free competition on workstations. This makes it desirable that open standards for the desktop should be introduced in the long term, particularly a file format for word-processed documents. An important question to be answered is what requirements should be set for a new standard in addition to being open.

The Apache developer Brian Behlendorf mentions a number of historical and cultural reasons why open source has been relatively weakly represented in desktop software.[14] Behlendorf points out that open-source developers in particular have worked on developing the form of software they used themselves, and that this did not include word processing suites and other office programs. In recent years, however, a number of good open-source programs for word processing etc. have emerged.

3.2.1. Open source alternatives

It is the impression of the working group that StarOffice and OpenOffice.org are the two open source alternatives in office software that are of greatest interest to government. The two products are largely identical. OpenOffice.org is a purely open-source product. StarOffice contains OpenOffice.org as well as a smaller, proprietary part, including database software. Licences for StarOffice are sold by the firm Sun for a few hundred Danish kroner. We have only concerned ourselves in the working group with the (predominant) part of the StarOffice which is also contained in OpenOffice.org, and we consider it reasonable to regard the two twin products as one – referred to in the remainder of this report as 'StarOffice/OpenOffice'. StarOffice/OpenOffice is run as a development project by Sun, and contributions come from both Sun and independent developers.

StarOffice/OpenOffice is interesting because it is generally recognised that the product is of high quality, partly on the basis of independent tests and reviews.[15] In addition, StarOffice/OpenOffice in our view is of particular interest because it:

- is a complete office suite equivalent to the programs in Microsoft Office, comprising among other things the word processing suite Writer (cf. Microsoft Word), the presentation program Impress (cf. PowerPoint) and the spreadsheet Calc (cf. Excel)
- uses XML as a basis for the product's own, preferred file format
- has good compatibility with Microsoft formats

[14] See Behlendorf's '*Open Source as a Business Strategy*' in the anthology M. Stone (et al.) 'Open Sources: Voices from the Open Source Revolution', O'Reilly 1999. This book is available online at http://www.oreilly.com /catalog/opensources/

[15] See for example PC World Danmark, August 2002 (http://www.pcworld.dk/default.asp?Mode=2&ArticleID=3620) or InfoWorld, May 2002 (http://www.infoworld.com/articles/ap/xml/02/06/03/020603apstaroffice.xml)

3.2.2. Test

The working group has undertaken a compatibility test of StarOffice/OpenOffice, i.e. of the ability of the product to exchange documents in doc format and other Microsoft formats. The test involved the product variant StarOffice and is described in an annex. The conclusion drawn from this test is that information is not normally lost on conversion (where loss means that part of the text disappears or changes), but that in some cases there is a loss of layout, particularly with regard to the location of layout elements.

It is difficult to decide whether the degree of difficulty of the documents in the text (e.g. the complexity of the layout) is representative of the documents exchanged between organisations in the government sector. It will be possible for most documents to be exchanged without problems, but in a minority of cases layout will be lost. A cautious estimate is that problems of some degree of seriousness will arise – most of which will be easy to overcome in 10-20% of documents. The compatibility problems which may occur in connection with decisions in individual organisations to switch to StarOffice/OpenOffice – while the rest of the government sector continues to use Microsoft formats – will therefore be surmountable, but will cause irritation and expense in the form of time spent on solving or circumventing problems with loss of layout.

It is difficult to imagine there being any open-source or other alternatives in the future that will be able to handle Microsoft formats completely free of errors. The doc format is a Microsoft business secret, and regular changes are made to it. Work on alternatives to achieve compatibility is therefore a constant uphill battle.

Overall, it can therefore be said that the public sector on the one hand benefits from there being a totally dominant standard. It is essential that documents can be exchanged without problems. On the other hand, competition between alternative office suites has to some extent been eliminated, because products other than those from Microsoft will inevitably face compatibility problems. This can therefore be described as a lock-in situation, which means that the dominance of Microsoft's products in the use of office software in government is maintained.

3.2.3. Open standards for electronic communication with the citizens

The question of open and closed formats on the desktop has another dimension with regard to communication between government and the public. As a matter of principle, it is unacceptable for citizens to be forced to choose office software, browser or other software from a particular supplier in order to be able to receive information from the public authorities.

According to rules laid down by the Danish National IT and Telecom Agency, government publications on the Net must conform to the HTML standard.[16] There is no direct reference to the principle of open standards in the description of the objective of the rules. A more pragmatic concept of 'technical broad availability' is used instead.[17]

Ministries, agencies etc. today publish a large quantity of information, which is readily available to the public. At the same time, it is difficult to comply with the policy of open standards (or broad technical availability) in external dealings with the public.

- Finished documents: the HTML documents published on government sector websites only fully comply with the HTML standard in a minority of cases.[18] The reason is often that the tools used to generate HTML codes automatically do not do so correctly, and that correcting the errors manually is time-consuming. In many cases the relationship is of no practical significance, but in some cases it creates difficulties for citizens who depend on the provisions made for the visually impaired in the standard
- Non-finished documents: it is common for the public to be obliged to have software that can handle Microsoft formats if they want to read more provisional documents, for example in connection with informal consultation procedures, or if they want to undertake electronic exchange of documents, for example with tax authorities. (This does not contradict the guidelines referred to above from the Danish National IT and Telecom Agency, which only apply to finished documents)
- Danish State Radio's service providing direct transmission of radio broadcasts over the Internet cannot be used by listeners who use the operating system Linux on their PCs, for example. DR has chosen Windows Media Player because it is free to the institution. There is an alternative product, 'RealPlayer', which can be used on Windows, Mac and Linux, but DR has to pay to use it. DR has been criticised by the open-source association SSLUG, and justifies its decision by pointing out that only 0.7% of DR's listeners use Linux.[19]

3.2.4. Opportunities and limitations in an open, XML-based standard for word-processed documents

The current difficulties in fulfilling the objective of open standards are serious in themselves. They are also an indication that with continued digitisation – which requires integration between a large number of systems and therefore a decision on which interfaces are to be used between the systems – the question of the use of Microsoft formats in particular will become increasingly pressing, both internally in government and in relation to the citizens.

[16] See the publication 'Om netpublikationer - Statens standard for elektronisk publicering', which is itself published on the Net in HTML, at the address http://www.netsteder.dk/publ/netpubl2/index.html

[17] See Chapter 2 of this report for a definition of the term open standard

[18] Hans Christian Studt's study of Danish websites, http://home13.inet.tele.dk/hcstudt/w3c/index.html

[19] Source: http://www.dr.dk/netradio/

The key area to try to influence with a strategy for the use of open formats on the workstation appears to be the file format government uses to exchange word-processed documents – at present the doc format from Microsoft Word.

Firstly, the format for word-processed documents is more central than the formats for the other facilities in office suites (spreadsheets, presentation programs etc.), simply because it is predominantly documents of this type that are exchanged. Secondly, it will be far easier for government to ensure that the external electronic exchange with the citizens takes place on the basis of open formats if government itself uses open formats in its internal exchange.

In the report's conclusions (Chapter 9), we draw up various scenarios for how government can introduce an open, XML-based standard for the exchange of word-processed documents. As a basis for the discussion on strategy in Chapter 9, we provide reasons in this section for some assumptions made for the discussion.

It is generally recognised as a matter of principle that e-government should be based on open standards. This is emphasised, for example, in the report 'Digital Forvaltning'.[20] The advantage of replacing the doc format with an open format in particular is that it will create real choice, with the consequence of greater competition on price and quality, and better conditions for choosing open-source alternatives.[21]

Another condition to be met is that government can continue to undertake internal exchange of documents without major compatibility problems. It is therefore important to try to define strategies that provide both benefits, i.e. supplier independence and a format that can be used by all public organisations.

In principle there is nothing to prevent standards for the exchange of office documents being open. The development of the Internet shows that open standards for extremely complex technical circumstances may be the basis for communication between programs supplied by independent producers (for example of web and mail servers and browsers). The present situation in the desktop area of 'standardisation by monopoly' is therefore not the only way of achieving standardisation.

The working group first enumerates some basic requirements which a format for word-processed documents has to fulfil. These requirements mean that existing, widespread formats such as HTML and PDF are not suitable. We then discuss the options for basing a standard on XML.

A standard for a format for word-processed documents has to fulfil the following requirements:

- The standard has to cover the layout of the document. It is not sufficient to be able to exchange texts without layout, for example as normally happens with ordinary electronic mail. Plain text (ASCII text) by definition does not fulfil the requirement of being able to reflect layout.
- The layout of the document must be independent of settings, type etc. of the software used to display the layout on a screen or print out the document. For example, a Ministry's logo must remain at the same place in the document. The HTML standard is therefore unsuitable, because the layout of an HTML document depends on the type and setting of the individual HTML browser, e.g. the size of the browser window
- The recipient of a document has to be able to modify the document. It therefore has to be possible to undertake electronic exchange of non-finished versions when cooperating on the formulation of a document. The format must therefore not be limited to being a final format. The PDF format is consequently not suitable.

The above requirements are far from exhaustive. A collective specification of requirements will be very extensive, as will the work on developing it, because it is not stipulated what it should cover. Among the other requirements a standard has to meet, word processing programs have to be able to save (store) a document quickly. The above requirements are emphasised because they show that the existing, widespread formats are not sufficient.

3.2.5. XML standard

On the other hand, there is much to suggest that XML (Extensible Mark-up Language) is well suited as a starting point for a standard for office documents. XML and associated standards are being developed by the World Wide Web Consortium.[22] XML is the basic standard, and the consortium is also working on the development of a range of associated standards, among other things to define transformations between different XML-based formats and to define layout.

XML is generally recognised from the outset as a valuable starting point for standardisation. In the report 'Digital Forvaltning'[23], XML is proposed as a key element in future standards for the exchange of data in government. XML has additionally been chosen as the starting point for some public standardisation projects, for instance in the healthcare sector[24] and in state administration.[25] XML and associated technologies are attractive, because they are based on open standards and because XML has already gained international momentum through the adoption of the principles of XML and the incipient application of the technology. It can therefore be expected that there will be a very large supply of software that supports XML.

It must be borne in mind, however, in the discussion on using XML as a starting point for a standard for word-processed documents, that it is a long way from the present agreement on the advantages of XML, purely in principle, to the point

[20] Ministry of Finance, May 2001

[21] cf. the discussion in 3.2.1 on compatibility problems in relation to StarOffice/OpenOffice.org

[22] http://www.w3c.org

[23] Ministry of Finance, May 2001

[24] as the basis for the development of electronic patient records, see http://www.sst.dk/faglige_omr/informatik/epj/med/medicin.asp

[25] The XML project on the exchange of data in the government sector, see http://www.oio.dk/XML/XMLprojektet

at which government actually has a usable standard. There are several reasons for this:

- XML and associated standards developed by the World Wide Web Consortium cannot *in themselves* constitute a standard for a suitable format. A standard of this kind can, on the other hand, be defined *with the aid of* XML and associated technologies. No such definition work is taking place under the auspices of the Consortium, and it also appears to be outside the Consortium's remit to work on such a specific application of XML
- The XML-related public projects under way in Denmark do not cover the development of an XML-based standard for word-processed documents. The projects referred to earlier in this section, on the other hand, deal with documents with some fixed structures (e.g. patient records and various forms), where an advanced layout is unimportant or of secondary importance
- Although XML and associated technologies are open, it is possible to *embed an existing closed standard in XML* without changing the closed nature of the original standard. For example, it is simple to define any document – including the fact that it is in a closed, binary form – so that it becomes what is referred to as a well-formed XML document (this can broadly be done by adding what is known as a 'tag' at the start and end of the document)
- Finally, XML in principle is geared towards the structure of data, not towards layout. The orientation of XML towards data relates to the fundamental idea behind XML of separating structure and presentation, in which XML differs from HTML. The purpose of the Word Wide Web Consortium's standard XLS-FO (Extensible Style-sheet Language Formatted Objects) is to supplement XML with a method to link a particular layout to a document, but it is still in a relatively early phase of development

The consequence of this is that before the government sector can introduce an XML-based standard for the exchange of word-processed documents, it is necessary either to choose an existing standard (which is therefore not 'official' in the sense developed by the World Wide Web consortium), or to enter into or implement a project to develop a standard. It does not appear reasonable to tie oneself to an XML-based format exclusively having to make use of the range of standards developed by the consortium, but to the extent that other standards are included it is very important that these are open, so that the complete document standard is open.

A candidate for an XML-based standard for word-processed documents is the one used by StarOffice/OpenOffice.[26] The standard is open in the basic sense, where the definition is publicly available, but also in the broader sense, in that there is a reference implementation of the standard with available source text (namely StarOffice/OpenOffice itself).

The working group cannot assess whether the StarOffice/OpenOffice format is appropriate (as well as being open itself), but some fundamental conditions apply in the format, which are general for an open, XML-based format developed in connection with a particular office suite.

On the one hand, as a result of the openness of the StarOffice/OpenOffice format there is a good starting point for competing suppliers to develop programs that convert to and from the format (known as filters). For example, Microsoft can develop a filter between the StarOffice/OpenOffice format and Microsoft's own closed formats, without having to use any form of 'reverse engineering'. On top of that, due to the widespread use of the XML standard there are some tools that facilitate the development of such conversion programs and, in the future, of programs that convert office documents stored in the StarOffice/OpenOffice format to future formats. Use of the StarOffice/OpenOffice format therefore does not introduce a lock-in and dependence on closed formats in the same way as when Microsoft's office programs are chosen.

On the other hand, it is true that if the format is established as a standard with the same dominance as the Microsoft format has today, StarOffice/OpenOffice will hold a preferential position in relation to competing products.

This preferential position (and therefore weakening of the competition between different suppliers) is due firstly to the fact that the format may be assumed to be adapted to particular features of StarOffice/OpenOffice. Secondly, the continued development of the development organisation behind StarOffice/OpenOffice is ultimately controlled in reality by Sun. This provides competitive advantages in being able to adapt the development of the format to the development of new features in the programs for word processing etc. in StarOffice/OpenOffice and in the form of best knowledge of new creations, questions of interpretation and so on in relation to the format.

3.2.6. Future format from Microsoft

Another possible candidate for an XML-based file format for word-processed documents is a future format from Microsoft. Representatives of Microsoft in Denmark have indicated to the working group that Microsoft's strategy is to compete on parameters such as functionality and quality, and not with the aid of closed formats, and that the firm is working on the development of a 100% open, XML-based format that can be used to save Word documents, in other words to be an alternative to the doc format. According to information from Microsoft, the idea is for the new format to be ready for use in conjunction with Office 11, which is due to succeed the current version of Microsoft's office package, Office XP.

The working group has tried to obtain more detailed information on the plans for the format. On the present basis, however, it is difficult to assess the planned format, including whether it will be able to contribute to increasing competition between different suppliers in the area of office software. Microsoft are unable to refer to any detailed descriptions of the format or plans for it, or

[26] see http://xml.openoffice.org/

an official commitment to developing such a format. Microsoft's representatives have stated that the idea is to develop a format equivalent to that which exists today for the spreadsheet program Excel. This format can be used to represent the basic content of a spreadsheet, while elements in a spreadsheet that are based on more advanced features in Excel cannot be represented in the format and are therefore lost if the spreadsheet document is saved (only) in the format.

In an assessment of these suggestions from Microsoft in Denmark, it can be said that Microsoft is already working with the XML standards in many areas. To take an example, Microsoft has been chosen together with the firm Accenture as supplier to what is known as the 'Inforstrukturbase project', which is part of the XML project mentioned above. An assessment must also take into account the fact that it is in Microsoft's interest to preserve its dominance in the area of office software, which is a legitimate business aim for Microsoft. It is not necessarily in Microsoft's interest to eliminate the compatibility problems associated with using alternative products.

In the conclusions (Chapter 9), we return to the question of what strategies government can choose to promote an XML-based standard for the file format for office software.

3.3 Infrastructure software

3.3.1. Open standards and supplier independence

The infrastructure area with web servers, mail servers, operating systems etc. is characterised in relation to open source as follows:

- The infrastructure is predominantly based on open standards, primarily the ranges of standards developed under the auspices of the Internet Engineering Task Force (IETF) and the Word Wide Web Consortium
- There is a high degree of competition and supplier independence. The infrastructure thus consists of a mixture of open-source and proprietary software, and in the main there are no compatibility problems between these
- There are many open source products in the infrastructure area that are recognised as being of high quality, including some that are market leaders (the web server Apache) and hold very large market shares (the operating systems Linux and FreeBSD)

In selecting software for a task within the area of infrastructure, it is possible in many cases to choose between solutions that are predominantly based on open-source software and solutions that are predominantly based on proprietary software. This is an extremely favourable situation for government and other customers investing in infrastructure, and the competition has also assisted in bringing down the cost of and spreading the infrastructure, and is consequently in a sense essential if it is to be possible to put the plans for e-government on the agenda.

One of the most important parameters government as a customer should emphasise in selecting infrastructure products is security.

3.3.2. Security in the infrastructure for e-government

The security aspect of software for e-government should be given extremely high priority. A high level of security is a prerequisite both directly for operation of the systems, including keeping them running at all times and at low cost, and indirectly for the public's confidence in the systems and therefore willingness to use them.

Security in software for e-government comprises protection against breaches of secrecy in the content of data communication (e.g. the economic circumstances of members of the public and companies and sensitive personal data) and protection against unauthorised access to computers (for example with the consequence of data being destroyed or a website hacked into so that it is defaced). The security requirements for software depend on the function and purpose of the software and are not limited to those mentioned here.

The problem of security in software for e-government is related to the open structure of the Internet. The fact that the Internet constitutes a link between all the computers connected to it – that is to say, many millions of computers throughout the world – on the one hand means that services are readily available and on the other results in vulnerability:

- The advantage of the Internet as an infrastructure for a large number of systems for e-government is that citizens and enterprises have ready access to the Net. It is easy to be connected to the Internet, and there is cheap and standardised software (including Internet browsers) to use it. Individuals and companies will therefore have easy and inexpensive access to services offered via the Internet
- The drawback is that public services offered on the Internet can be made the object of attack from any computer connected to the Internet

3.3.3. Security and form

The following sections discuss the relationship between security and the form of software licence.

First of all, it can be noted that neither the open source form nor the proprietary form contains a guarantee of the software having a high degree of security. The same applies to quality in general, including user-friendliness. The advance of the open source form in relation to security is the availability of the source text, which makes it possible for everyone – including independent experts – to contribute to security analyses, tests and the fixing of bugs. The advantage of the proprietary form is that the sale of licences makes it possible to finance such activities.

There are open source products that are generally recognised as offering a very high degree of security, such as the web server Apache. There are also open source products plagued by bugs, such as the BIND program, which is a widespread implementation of the DNS protocol. (The DNS protocol is the basis of translating between symbolic and numeric Internet addresses).[27]

[27] Source with regard to the BIND program: Rik Farrow: Security of Open-source software, September column

3.3.4. Example: Security in the web server Apache and Microsoft's Internet Information Server

The security features of the web server are of very great significance in an infrastructure for e-government that to a great extent is based on the Internet. A web server receives and processes queries sent from a web browser. The queries are processed by obtaining and sending the queried document to the web browser. The document may be an html page or pdf document, which already exists (e.g. an organisation's website) or a document created from time to time on the basis of the individual query (e.g. a query arising when the user uses a search function on a website).

The web server has to ensure compliance with more detailed rules laid down for which documents may be sent. A web server often has to be able to authenticate browsers (which may, for example, be authorised to make particular enquiries about confidential material) and assist towards data communication being kept secret by encryption. In a broader sense, the web server has to protect against a number of different forms of attack, including denial-of-service attack, unauthorised modification of documents and programs on the web server and unauthorised execution of programs. The web server is exposed to these and other forms of attack, because it can be accessible from computers throughout the Internet.

Web servers implement the server part of the http protocol. The http protocol is an open standard, and there are a number of competing products, all of which have the same basic functionality. Two of them are the open source product Apache and Microsoft's Internet Information Server.

3.3.5. Attacks by the worms Code Red and Nimda on Microsoft's Internet Information Server

In 2001, web servers throughout the world, including Denmark, were attacked by the computer worms Code Red and Nimda. A worm is a program that spreads automatically from one attacked machine to another. Both worms made use of some security bugs in Microsoft's Internet Information Server (IIS) Versions 4.0 and 5.0. The attacks led the large, independent IT consultancy Gartner Group to recommend that affected companies should replace IIS with one of the alternatives, and specifically mentioned the web servers Apache and iPlanet (the latter is proprietary software and is now sold under the name of Sun One Web Server).[28]

Code Red is estimated to have affected several hundred thousand web servers. On some of the web servers infected with the worm, all replies to browser queries were changed so that they displayed the text 'HELLO! Welcome to http://www.worm.com! Hacked By Chinese!' In addition, the worm from the affected machines made a denial-of-service attack on the web server in the White House in Washington. The Nimda worm did not give visible expression to itself in the form of website defacement, but caused problems by infecting the web servers with an alien program and created a large amount of extra data traffic when it tried to spread. These worms cause huge costs in terms of working time spent clearing the affected computers of the worms. Both worms were particularly advanced, and attempted to exploit a large number of security bugs. Nimda exploited bugs in both IIS and Microsoft's browser Internet Explorer and also spread through e-mail.

The more specific problem in connection with the Code Red and Nimda worms is that they both exploited security bugs that were known beforehand, and that the web servers would have been protected against if their system administrators had installed the updates (also known as 'patches') that had already been sent out by Microsoft.

The Gartner Group's criticism of Microsoft's IIS was that the quantity of security-related bugs was so great that they led to a totally unacceptably large number of regular, bug-fixing patches from Microsoft. It was claimed in the statement from Gartner Group that Microsoft sent out patches almost every week, and that this was associated with unacceptably high costs in maintaining the IIS server, in installing the regular patches and in other ways. Gartner Group anticipated that serious security bugs would continue to be found in the web server until Microsoft developed a new, completely revised IIS.

Microsoft's response pointed out that serious security-related bugs have existed in all web servers and platforms for web servers, and that it is essential for all organisations with security-critical systems to develop procedures that ensure quick installation of all necessary patches.[29]

3.3.6. Comparison with Apache

It is difficult to compare the security in different products, even if the products have the same function, such as web servers. This is partly due to the fact that there are no collective statistics on how many attacks there have been on the various product alternatives, and it is difficult to imagine how statistical material of this kind could be collected. It would also have been very difficult to define a yardstick that can cover important and more specific parameters than just the number of attacks, for example the costs associated with repairing the damage they cause. On the other hand, it is possible to enumerate the number of reported bugs and associated fixes for the various products, but the problem here is that these bugs cannot be directly compared, partly because their possible consequences – that is, the scope of the attacks that potentially could be made by exploiting the bugs – depend on the features of the complete software, on which it may be difficult to obtain an overview. Finally, it is even more difficult to assess the risk of new bugs being found in the future. This is partly due to the fact that a large number of bugs found may be interpreted as a risk that further bugs exist but may also be an indication of an effective test procedure that has substantially reduced the number of unknown bugs.

Despite the difficulties in comparing the security of different products, in the view of the

in Net Work Magazine, http://www.networkmagazine.com/

[28] Source: Gartner FirstTake 19. September 2001 by John Pescatore, http://www.gartner.com/resources/101000/101034/101034.pdf

[29] Microsoft's response was discussed by many web-based news services, including that of ComputerWorld at http://www.computerworld.com/securitytopics/security/story/0,10801,64226,00.html

working group there is a great deal of evidence that the open-source web server Apache provides a high level of security, and that the security is higher than with the proprietary product Microsoft IIS. We justify this statement firstly on the basis of the good security history of Apache with regard to the number and nature of security bugs found. A serious security bug can be defined as a bug that potentially gives an external attacker the opportunity to have any chosen program code executed on the web server.[30] Apache's history is that after a report in January 1998 no bugs of this type were found in 1998-2001. During 2002, bugs of this type have again been found.[31]

Secondly, the whole Apache program is executed with rights equivalent to an ordinary, non-privileged user, while parts of IIS are executed with far-reaching administrator rights. The higher rights the program is executed with, the greater the damage that can be wrought in an attack of the type on which Code Red and Nimda were based. This is due to the access of the hostile code to files and other resources on the machine being dependent on the web server's own level of rights.

Thirdly, the source text has been available for several years and has been the subject of extensive reviews and analyses. We discuss the significance of this in the next section.

The security aspect of software on workstations is also significant for the general security in the infrastructure. As mentioned earlier, the Nimda worm also spread by exploiting bugs in Microsoft's browser Internet Explorer.

It is the working group's general impression that Microsoft's software on the desktop has also been plagued by constant, major security problems. Quite recently, a new security problem has been revealed in the operating system Windows 2000, of the serious type where a hacker can cause any chosen code to be executed on the workstation. According to an article in ComputerWorld on 2 August 2002, the bug has been eliminated with a fix that had just been released (in the form of what is known as a service pack) from Microsoft, but at that time did not exist in a version that could be installed in Danish-language editions of Windows 2000. Danish desktops with Windows 2000 are vulnerable to such an attack if the security level is selected as 'medium'. As an interim solution, the problem can be solved if the user is aware of it and is able to change the security setting to 'high'.[32]

3.3.7. Security and complexity of software

It is possible, in principle, to design software that provides a very high degree of security using encryption systems, digital signature, firewalls and other methods and principles. Experience shows, however, that it is very difficult to design secure software. This is evident from the fact that, despite large resources being invested in software development and security normally being given high priority, new security-related bugs are regularly discovered – whether it is open-source or proprietary software.

It can be said from a general perspective that one of the main causes of security bugs is the complexity of the software. The software has many functions, and consists of thousands and in some cases millions of program lines, so that it is not possible for a single person to have an overview of the software, and there is a complex interaction between its various constituent parts.

Software development requires extensive resources in the form of labour for design, programming and testing/fixing. In the case of both open-source and proprietary software, it is essential that such resources are actually used if a high degree of security is to be attained. When we talk here about development, it is meant in the broadest sense, both the first finished version of a software product and the enhancement (or maintenance) of the product in the form of the work on creating new versions with more features and so on.

We divide the discussion of the various conditions for allocating resources for the development of secure software in open source and traditional projects into two parts: the significance of the available source text (which invites reviews) and what may be termed the development model, where a characteristic feature in open source development is often the participation of many (but where there are no resources for reviews).

3.3.8. Security and availability of source text

The advantage of open-source software in relation to security is the availability of the source code. The advantage consists in the fact that it is possible for everyone to analyse the software, both at an overall architectural level and with regard to the individual parts of the source code. This provides great potential to identify and eliminate bugs. It also enables the user to obtain credible, independent assessments of the software. This also applies to the assessment of a claim regarding a current, disputed bug, as it will be difficult for a supplier to deny a bug if this is documented with reference to the source text.

The publication of source text for security-related software obviously means that the source code is also available to potential hackers and so on. It is a generally recognised principle, however, that security in software must not be conditional on the source text of the software being kept secret, the encryption methods used etc. The secret element must only consist of encryption keys, passwords and similar items. Rules and guidelines for use of the systems must ensure that the various keys are only known to relevant people or programs. If, on the other hand, security is conditional on the actual method used or its implementation being kept secret, unacceptable dependence on such secrets being preserved is introduced, for example a dependence on individual people and organisations who have been involved in the development of the software or who are involved in its operation and maintenance.

Examples of publicly known security-related algorithms and methods are the principles of public key systems (Public Key Infrastructure) and the DES algorithm (Data Encryption Standard). A system based on public keys has to constitute one of the

[30] It was this type of bug that made possible Code Red and Nimda, as the worms were program code that was transferred to the web server from outside and was then executed on the web server

[31] Sources relating to security defects in Apache in 2002: http://www.apacheweek.com/features/security-13 and http://www.apacheweek.com/issues/02-03-01

[32] http://www.computerworld.dk/default.asp?Mode=2&ArticleID=15536

basic elements of e-government. The DES algorithm has been widely used for the encryption of data communication, in recent years in a variant (also publicly known) under the name of triple-DES. The American National Institute of Standards and Technology (NIST) has selected a new algorithm by the name of Rijndal as the successor to DES and triple-DES. The Rijndal algorithm has been published, and NIST has selected it among a number of candidates, the advantages and drawbacks of which have been discussed by security experts quite openly. In the case of all these methods, the method itself is public, and the secret element consists only of the encryption keys.

In the same way that the publication of an encryption method is not a guarantee of the method being good, but is merely an element in a process in which the method is illustrated, publication of source text for security-related software is not in any way a guarantee of security.

Although the security of a particular software product must not be conditional on the methods used and the actual implementation in source code being kept secret, publication may make it easier for a hacker to find the weak spots in the software. The ideal, but impossible situation, is for the source code to be available to those who want to analyse it to fix bugs, and unavailable to those who want to exploit the bugs.

The overall impression of the working group is that where open-source software that has been widely used for several years and has been the object of extensive analysis is involved, and if this process has led to positive assessments by independent experts, this provides the user with an extra degree of security in comparison with proprietary software. The supplier of traditional software can provide selected, independent experts and institutions with access to the source text. There may be increased confidence in the software, but in the nature of things, the process will always be limited to covering a selected circle, and it is not general.

3.3.9. Security: simple bugs versus subtle bugs

In the model for proprietary software, the sale of licences for the software can finance the allocation of resources to systematic testing and fixing of bugs. This raises the question of what mechanisms exist in the open source model to promote allocation of the necessary resources in the form of manpower for development, testing etc.

3.3.10. The bazaar model

With a view to illustrating the difference between resource allocation for software developed in open-source and traditional projects, we shall take as our point of departure Eric S Raymond's 'bazaar model'. In his essay 'The Cathedral and the Bazaar'[33], Raymond has established a model of how software development proceeds in an open-source project. The model is interesting partly because it describes incentives to contribute voluntarily to open-source projects.

A key element in Raymond's bazaar model is that users contribute their own manpower, because they themselves have an interest in the software being improved. Raymond goes on to characterise the projects as follows:

- 'Release early, release often' – under this principle, the software is made available in development versions for a large number of people, making it possible for a large number of users to contribute to finding and fixing bugs
- 'All bugs are shallow' – this is based on an assumption that bugs in the software typically are trivial; the decisive factor is that a large number of users assist in testing the software

Alongside the user's own interest in improving the software, Raymond uses the term 'ego boosting' to describe a further incentive to contribute to software development.

Raymond's bazaar model on the one hand encompasses some key mechanisms in open source projects based on the voluntary participation of people who are both users and developers. In many cases these are users who are particularly skilled software developers with great expertise in the area in which the software is applied, and whose use of the software may be both private and commercial. An example of the user/developer dual role of the open source model is the basis of the Apache group in people who worked on some of the early websites as system administrators and in other similar roles. They had a commercial interest in the software being developed, and they had high levels of skill with which to contribute. Alongside the possibility of attaining some form of honour or recognition, the participation of individual people may also be linked to a financial interest in relation to job opportunities, if they build up and demonstrate high specialist know-how by participating. This form of self-interest is supported by the source text being available together with an indication of who has written it.

On the other hand, Raymond's description points to a weakness, namely with respect to the need for experts to systematically examine the software with a view to finding non-trivial bugs that require special understanding, for example, of security issues. An attempt can be made to take account of this in an open-source project by organising various forms of systematic reviews of the software. The basic model with voluntary manpower motivated by self-interest in improving the software does not, however, directly accommodate a mechanism that ensures that such activities are actually carried out. It follows that considerable time and special knowledge are required in relation to the gain made by the individual when the software is improved.

3.3.11. Proprietary bug-fixing

It is essential in the model for proprietary software, where the sale of licences to the software can finance allocation of resources for systematic testing and bug-fixing, that:

- the testing activities to some extent take place independently of the development activities
- and the testing activities are given high priority regardless of the situation that the time spent on them may conflict with the economic interest of the company in a quick launch of the product/product version

[33] Source: http://www.tuxedo.org/~esr/writings/cathedral-bazaar/

An example of a highly forced development is Microsoft's efforts in the mid-nineties to enter the market for web software and browsers. Microsoft did not recognise the significance of the Internet until late on, and the consequent time pressure has presumably contributed to the constant security problems of Microsoft products in this area. The difference between the forces driving the (quick) launching of open-source and proprietary software is discussed more fully in Chapter 4.

The testing activities of a software development house should be independent of the development activities in the sense that the objective for testing activities has to be to find as many bugs as possible, rather than attempting to show that there are few bugs. This can be promoted by organisationally placing the testing activities in separate units with their own objectives and with no staff who are also involved in the units developing the software.

Overall, it is not possible in the view of the working group to say at a general level that software developed according to one model has a higher level of security than software developed according to another. The open-source model makes it possible to involve a very large number of people in testing and bug-fixing activities and for the user/purchaser to have access to independent assessments; and there is no incentive in the same way as with proprietary software to force the launch of a product and in so doing shorten the time spent on testing and bug-fixing activities. The proprietary model provides more opportunity to allocate paid manpower to testing and bug-fixing activities, and although the user/purchaser normally does not have access to independent assessments in the same way as in the open source model, it is possible to organise testing activities internally in a software development house so that their objective is actually to find as many bugs as possible.

3.4. Custom built software

Custom built software covers a wide range, and encompasses for instance:

- highly standardised systems that fulfil a specific function (pay, financial control etc.), for which there is a need in many public organisations, regardless of their administrative tasks
- systems that cover the same needs within a group of public institutions of the same type (e.g. electronic patient records in the healthcare sector)- here called function specific systems.
- highly specialised functions that exist in very few places or in only one place (e.g. the Central Customs and Tax Administration, the central population register)

The characteristic feature of custom built software is that it does not exist in advance, but is developed by a supplier on the basis of a specification of requirements in a public invitation to tender. If the software is to be supplied as open source, this is to form part of the specification of requirements.

3.4.1. Custom built software as open source

The fundamental consideration in connection with requiring custom built software to be supplied as open source will in practice be whether the requirement entails an increase in costs, because the supplier loses the option of selling licences for the same software to other supplier. This problem is analysed more closely in Chapter 7.

The requirement to supply custom built software as open source need not be an either/or situation but can be specified as being concerned with the adjustments a supplier makes to existing standard modules, if these are proprietary.

The requirements of the public sector for rights to software for the development of which it has paid can also be varied so that the public organisation has access to the source code and can modify it (which provides better prospects of being able to compete on tasks relating to the maintenance of software) but does not have access to distribution of the software to other organisations with the same software needs (so that the supplier does not lose the option of selling licences to these organisations). The Danish Competition Authority has therefore recommended in 'Konkurrenceredegørelse 2000'[34] that purchasers of software stipulate a number of requirements in connection with the purchasing of software (referred to in the Authority's report as 'adapted programs') assure themselves of as many rights to the system as possible. The Authority recommends in general that purchasers of software are aware of the question of rights, and also mentions a number of less far-reaching rights to the software that should be ensured, including the right to outsource operation and the right to replace the hardware the software is run on.

3.5. Conclusions

The conditions for increased application of open-source software on the desktop are governed by the fact that Microsoft's closed file formats are dominant. There are good open source product in office software, but although StarOffice/OpenOffice, for instance, has attained a high degree of compatibility with the Microsoft formats, the alternatives face an uphill battle to attain complete compatibility, because the format can be changed by Microsoft. The consequent lack of competition is doubly serious in that there is a trend towards the use of proprietary formats also spreading, albeit to a far lesser extent, to electronic communication with the public. The most essential condition to be met for increased application of open source on the desktop, and altogether to establish greater competition in the area, is that the public sector makes sure that word-processed documents are exchanged in an open file format, and there is much in favour of the format being based on the XML standard.

There are good prospects of using open-source software in the area of infrastructure software, because the area is dominated by open standards and there are established open-source products recognised as being of high quality. Very high priority should be given to the issue of security in choosing between different solutions in the area of infrastructure. The licensing of software either as open source or proprietary does not automatically

[34] (http://www.ks.dk/publikationer/2000/kr2000/index.html)

guarantee a high level of security. But security in open-source software can be made the object of independent reviews, because the source code is available. And if the software has existed for a prolonged period of time, the source code has been the object of extensive and independent analyses and these lead to a positive assessment, as in the case of the open-source web server Apache, then there can be a greater degree of confidence in the open-source software than in proprietary software, where the source codes is not normally subjected to independent reviews.

Apache may be assumed to offer a higher degree of security than Microsoft's web server Internet Information Server, which in 2001 was attacked throughout the world by the 'worms' Code Red and Nimda, resulting in great inconvenience and cost. Security in the infrastructure is also dependent on the security of workstations in government, where Microsoft's software has also been plagued by significant security problems.

In public invitations to tender for tasks relating to custom built software, requirements that the whole or parts of the software are supplied as open source can be specified. Consideration can also be given to setting requirements for less far-reaching rights, such as access to the source code and the right to modify it, which provides the important option of choosing between different suppliers to deal with maintenance.

Chapter 4

Economic analyses of open source

This chapter analyses the economic difference between using open-source software and proprietary software. The analysis is based firstly on rights of use and secondly on program development and maintenance, referred to in brief as development, including the question of upgrading. The chapter ends with a cost model for use in assessing open-source versus proprietary software.

4.1. Economic characteristics of software as a product on a market

From the economic point of view, choice of software is equivalent to other types of investments, in that the acquired asset provides a return by creating value over a longer period. At the same time, acquisition of a program is an option that is irreversible, uncertain and takes place in a world undergoing significant technological development, which may lead to the emergence of far better programs. The economic analysis is based on differences between open-source software and proprietary software consisting firstly in differences in the economics of right of use and secondly in differences in the economics of the way in which the software is used, including program development and maintenance, referred to in brief as development. To be able to draw up relevant models for an investment of this kind, it is necessary first to clarify features of software as a financial asset.

The financial investment in software is to a great extent tied to the right of use, which entails a number of financial differences between open-source and proprietary software. This is due to the special circumstances in the production of software. Firstly, programs do not have any physical substance, and consist solely of information, which also means that original and copy are concepts devoid of meaning. Programs can be stored on various media, and the choice of medium is a matter of habit, technology and economics. This also means that there is no physical wear on the software during its use, and there is therefore no physical limit on its useful life. All in all, this means that programs do not have any value as physical products, but have value in the right to use the product. We therefore apply the perspective of property rights in the financial analysis in this report.

Secondly, the development costs for software are quite considerable, whereas the marginal costs in copying and distribution are insignificant, particularly where only sale of the right of use is concerned. Particularly for suppliers of software with a large number of units sold, this means that the supplier can set a price that bears no relation to the costs of development and is solely based on the value of the right of use. As different users have different needs, this means that the ideal price is based on the customer's willingness to pay and that software is therefore sold with sharp differentiation of price. The price level may vary over time and in different markets and may show opposite trends, i.e. while the price rises in some markets it falls in others due to competitive considerations. Only strong competition between alternative products can ensure that a downward price trend can become established.

The economics in the sale of software do not follow a classic pattern, because the profit on the last unit sold is largely equal to the sale price. At the same time, the producer does not have any capacity limits, and the size of the market constitutes the only limit on sales. There will therefore be a trend towards the formation of monopolies in standard software, and it will therefore be difficult for competition to be maintained by allowing market forces to dictate development.

Finally, there have been many innovations in software products, which have redefined the prospects of competition in both hardware and

software markets, network technologies, WWW etc. Innovations have undermined dominant market positions in some – but not all – cases by creating new competitive conditions.

4.2. Limitations of the analysis

Organisational circumstances have a strong bearing on efficiency in the utilisation of programs. Where differences between open-source software and proprietary software lead to essentially different organisational requirements for maintenance and updates etc., it is difficult to state the economic differences comparably. We have chosen to focus on examples where we can look at the economics of functionally comparable programs. It is also assumed that programs are acquired on a comparable scale and on a given platform, which means that we look exclusively at software. We therefore avoid entering into a discussion on alternative architectures, while being fully aware that such a choice may have far greater economic consequences than the choice between open-source and proprietary software. Similarly, we assume in our examples that quality is comparable, i.e. we assume that the users consider their needs to be met to an equivalent extent.

Both the number of open-source software products of relevance to e-government and variations in the environment in which they are embedded are substantially greater than shown in the examples in the following chapters. The conclusions are therefore to be regarded as providing a guide for considerations in other contexts, as the value of open-source software products with regard to functionality and requirements for maintenance etc. must be specifically assessed in comparison with proprietary software products. The results of the studies presented here are generalised, while emphasising the reservations that need to be made.

As this report is geared towards administration, we disregard the significance of open-source software for commercial enterprises in this report, which means that there may be other considerations over and beyond those mentioned here on which a private enterprise wishes to base its software acquisitions. The common denominator for open-source software is the specially formulated *rights* to the source code in a program, as already mentioned in Chapter 2. The economic consequences of this are discussed in the next section.

4.3. The economic rights perspective

Unlimited access to using copies of an open-source software program implies an immediate benefit for users of open-source software, as the costs of right of use are independent of the number of users, when compared with proprietary software, where the right of use is paid for as a sum per user, although licence price often falls with the number of users. At the same time, the rights also lay down the options for making changes to software or integrating open-source software with proprietary or customer-owned software. When comparable products exist, this *rights perspective* therefore points to direct and clear economic advantages in acquiring open-source software rather than proprietary software products.

The formulation of rights also has an impact on economic differences in the longer-term perspective. This relates, for example, to free choice of supplier for maintenance, and in certain circumstances even free access to maintenance without payment, adaptation to future program procurements, free access to choice of supplier of integration software and so on.

When there are no directly comparable products, the economics of the design of open-source software and proprietary software depends on how specifically the product is designed. A software product that can only be used by the customer concerned ('organisation-specific software') cannot justify price differences between open-source and proprietary software, as all the development costs have to be covered by this one sale. An open source licence for a software product with several/many potential users will mean a potential loss of earnings for the developer. This applies both to specific software and to general software such as infrastructure software or desktop software. This could directly justify a significantly higher procurement price in relation to the first customer, but competition for the contract will reduce a difference of this kind, as the winner of the contract will be left with software skills of value, including those for potential customers for the product, and with relatively better prospects of gaining the development contracts (installation, adaptation, enhancement) that are essential if efficient organisational utilisation is to be achieved.

The extent of the economic differences between open-source and proprietary software depends on other factors in addition to those discussed above, as prices and terms of licence for proprietary software are altered and adapted to market conditions, so that differences between open-source and proprietary software vary over time under the influence of market conditions.

Suppliers with their own software products have responded in widely differing ways to the penetration of their markets by open-source software products. Responses range from recognition and acceptance, with the inclusion of their own resources to maintain and develop a product, to pure rejection and suspicion. There are wide differences in the price strategies adopted by different competitors in relation to open-source software, depending on market shares and potential for the individual open-source software products. In other words, we cannot talk of open-source software as a family of products where exactly the same economic advantages (and drawbacks) apply to all products, as these products have been viewed in the light of the alternative options among proprietary software on the market. We therefore restrict ourselves in the analyses that follow in Chapters 5-7 to a small number of comparable products.

4.4. The economic development perspective

If the right of use is to be of value, the software has to be implemented, procedures changed, users trained and so on. Both this short-term change in the organisation and the more long-term change in the organisation in conjunction with renewal and replacement of software form the basis for the analysis of differences between open-source software and proprietary software. This is referred to overall as the development perspective.

For proprietary software, development and maintenance are only done by the owner of the copyright to the program. Responsibility for

development and maintenance for open-source software may rest with both the original holder of rights (who stipulates that open source rights are to apply from then on) and the individual user, as any user in principle can enhance the product and customise it to his or her own wishes, while complying with the open source rights that apply in this situation (see Chapter 2). User-as-developer-and-tester is the decisive difference from proprietary software, as open-source software via user groups on the Internet has the potential for rapid and versatile, mutual support with free development of software improvements and corrections, i.e. without obstructive software rights.

The development perspective covers both centralised and highly decentralised development concepts for open-source software products, which affects the time perspective, skills requirements and organisation. The time perspective for an economic asset is of decisive importance to its value. It has been noted that for every Danish krone spent on acquisition of IT, ten kroner is spent on organisational adaptation, development etc.[35] The economic life-span of software must therefore not just be compared with the procurement price. When the other costs are added, the requirement for software life increases to include cost-effectiveness, in the same way that replacing existing software makes greater demands on software product improvements than are equivalent to the additional price for new purchase. Where these improvements and their (expected) productivity effects cannot be documented, a change for other reasons must therefore of necessity be linked to a very high degree of functional and user-interface agreement with the software used to date in order to minimise installation, maintenance and training costs. These other reasons may be a revision necessitated by bugs and deficiencies in the software.

The difference between internal IT skills being available and external consultants with relevant skills having to be called in gives rise to differences in the structure of the operating costs as well as that of the organisation, which may make comparisons between solutions in the short term difficult. It is only reasonable to compare different solutions if a multi-year perspective is adopted.

Disregarding the development perspective would mean *overlooking* the very substantial difference between open-source software and proprietary software: unobstructed entitlement to adapt and expand open-source software products locally in an organisation, regionally in an authority area or nationally for a public institution. The next section describes software investments as options.

4.5. Options

Options (see footnote) form the key to understanding the economics in the use of software in a development perspective,[36] reflecting the fact that an investment has to be based on the principle that investments are generally irrevocable, uncertain, and have timing problems, so that it is necessary to bring the time of the investment into the overall considerations. In the case of real options, a project (the investment) is identified and a decision-maker assesses how its premises and potential are developing before the investment is made, while the classic method is a financial assessment comparing the expected return from a project with that from financial securities. Software investments as a rule entail follow-on investments, the value of which is significant for the original investment value. Upgrades can be regarded as a follow-on investment where purchasers of previous versions obtain the new version more cheaply than new purchasers. We shall look at three aspects of software as real options: irreversibility, uncertainty and upgrading.

Irreversible investments are investments that cannot be cancelled without financial loss. Investments of this kind therefore have far greater consequences than those linked to the narrower rights perspective. The degree of irreversibility of an investment can be assessed on the basis of the value of the real option expressed as 'switching costs'.

The second factor in real options is *uncertainty*. In relation to software, this factor applies constantly during the time a software product is in use, as a whole series of requirements of development opportunities have to be met for the acquired software to fulfil the expectations on productivity gains. Technological innovations can reduce the value of the acquired software if new software offers substantially improved functions. New requirements for interaction with other software can reduce the functionality of previously acquired software. There will not be certainty about any of these changes in the procurement of software, but the history of software shows that there will be technological innovations. By offering integrated software products, the market has tried to reduce uncertainty over the functional value of the product, but has increased the requirement regarding organisational adaptation to the product concerned and in so doing has increased the organisational uncertainty over whether the organisation is capable of harvesting the functional value.

Upgrading is a special problem in assessing software as an investment. From the point of view of the investor, the economic life-span is limited by the emergence of better software, where added value of the new software is greater than with the present software. In other words, software is not to be replaced until the time when the value of future use (including purchase) of a new program is greater than the value of continued use of the existing one. From the producer's point of view, upgrading is an economic necessity, as sales opportunities for unchanged software are expected to be reduced for every licence sold (not because of pirate copying but because of declining novelty value and the emergence of competing products). The producer therefore has a strong incentive to continue to offer new facilities. The greatest competition, particularly for products with great

[35] Brynjolffson 2001, on Matrix of Change, MIT

[36] An option is an agreement that provides a right, but not a duty, to implement the contents of the agreement. If the expected value of the agreement is negative, the value of the option is 0, because the agreement will not then be implemented. If the expected value is positive, it is the value of the option. The value of an option can be calculated from a probability distribution of expectations for the future.

market coverage, comes from earlier versions of the same product.

The option elements of software products depend upon the moment of investment; external influence like changing software technology as well as internal IT organisational capacity and competences vary over time. What is a good time and a good choice of software are determined by circumstances that are uncertain, because options are an assessment of future circumstances that are not only under the investor's own control. A more detailed account will be given below of the costs associated with software in a development perspective, involving the options perspective that has been outlined above.

4.6. Software costs in an options perspective

In the following sections, we illustrate economic features of software of significance to an understanding of differences between open-source and proprietary software. The analyses focus on the 'life cycle' of software in order to be able to illustrate the consequences in terms of option economics of software application for the user. Table 4.1 presents the economic characteristics of options in open source as opposed to proprietary software.

Table 4.1. Factors for assessment of options in investment in software products

Options feature of software products	Definition	Option value	Open source v. proprietary software
Specificity	Every piece of software imposes a set of requirements for its hardware and software environment	Choice of software entails some degree of lock-in, i.e. irrefutable loss in changing to an alternative which is greater the more specific the software in its requirements for its environment	Both types may entail lock-in, depending on how specific the software is in relation to application
User learning curve	Software is a knowledge-based product with a period of learning for users.	Losses on training costs are accompanied by indirect losses in reduced production during learning time compared with the situation with full experience	The learning curve depends on the user-friendliness of the product and not on the type of software
Support learning curve (conditions to be met for maintenance and support)	Skills of IT department in supporting users and maintaining software	Specific investment in staff and software tools that cannot be fully re-used but entail loss on change of software	Proprietary software often has supplier-specified training. The content of this training sets the boundary between the service that the customer, or certified consultants, provide, and the service provided by the supplier. Open-source software leaves it to the customer to establish who does what
Compatibility	How capable a software product is of cooperating with other software	Complementarity effect (positive economic value) and opposite (if there are requirements for conversion etc.)	To the extent that proprietary software uses closed interfaces and closed data formats, there will be 'lock-in'. It will always be possible to read interfaces and data formats in open-source software. It can therefore be made to be compatible with other products
Integrability	Every software environment needs to be able to incorporate new software efficiently, and software products may make this more or less efficient	The costs of software integration rise in line with the incompatibility of software products and increase barriers to the acquisition of new software	The integrability of proprietary software depends on the supplier's product strategy in relation to integrability. Integrability for open source depends on the willingness of the programmers to supply integration. Alternatively, users can integrate the products themselves

Behind the direct user interface there is a whole series of other programs that are decisive to the possible uses and stability of the overall system, as all these layers of software make the necessary facilities available to the user. The individual programs can therefore never be viewed as an isolated investment, but have to be viewed in connection with the base already installed. The environment of any program is formed by other programs, some of which are present as software, while others are encoded into electronic equipment, such as monitors, printers, scanners, digital cameras etc. This means that replacement of a software product cannot be assessed independently of ties to other programs (e.g. operating systems, network protocols etc.) and independently of ties to hardware (e.g. requirements from Intel's microprocessors, IBM's RISC processors, etc.) and the capacity characteristics of hardware. From the economic point of view, this means that there is interdependence, so that freedom to choose technology is precluded, and we therefore talk of a degree of 'lock-in', which imposes on the customer special costs that cannot be offset.

Over and above these technological conditions, the user gains knowledge and experience in the use of software, which means gradually increasing productivity (all other things being equal). If this experience is not relevant in new software, there will be a loss of human capital (assets) when new software is purchased. In addition, new software entails a new induction period. The economic costs of teaching staff a new 'tool' are not limited to the *direct* cost of doing so, but also include the indirect *loss of production* from a decline in productivity during the running-in or learning period. There is also a lock-in effect, the total value of which rises with the number of users.

A third feature of software is that data from it can be used by other software. A program therefore has to be able to talk to a program on the equipment of customers or partners in cooperation. Standards, conversion programs and exchange formats have been created for this purpose that can be present as expansions to the individual program or be present on servers with special conversion equipment, which is guided by 'who' the data concerned (the individual file) is intended for. From the economic point of view, these programs increase the value of the 'core program' by increasing the usability of its data. The technical term for these features is protocol and data compatibility and network 'connectivity' or interoperability. We talk of economic complementarity, which means that the value of a software product grows with increased use of another product, which also creates value.

Finally, programs are not completely self-instructive or self-explanatory for the human user. Nor are there any bug-free programs – regardless of what is claimed – when the program is expected to work in many different program environments. Every program therefore requires maintenance and support to work satisfactorily for the user. The maintenance and support capacity and skills of the organisation therefore have to be included when program replacement is considered. The maintenance and support function includes a special program to monitor and debug other programs, which can be done with custom built software for the program that is procured. These are operating costs, which track the investment in a program, and which may have limited use in another software development, so that investments are of little value in alternative application.

The operation of a software environment takes place with a chosen level of service quality (for example measured as the time from when bugs appear until bugs are dealt with at the user's site, down time during the course of a year, time from breakdown to restart, time set aside for user instruction, time for access to backup etc.), which is significant for the requirements of skills and the number of IT personnel locally and/or of service agreements with suppliers. Service costs are a function of service level but also of what entitlement there is to troubleshoot a program. Finally, service quality is also normally a function of experience, so that there is also a learning curve here.

Every program has limitations. There is therefore reason in the longer term to expect a need for *new* programs, which provide the organisation with new opportunities and solutions. Integrating a new program into the portfolio of an existing organisation is just as essential as being able to maintain the program already installed. This applies regardless of whether the reason for newly acquiring a program is internal development of the organisation or whether it is due to pressure from a new market supply that sets the agenda for the service and production of the organisation. Software that can be more easily incorporated into an existing program portfolio than alternatives has lower installation costs, and the organisation therefore has a lower barrier to innovation than others.

All these factors are of key significance as they each separately contribute to the investment option value of software. When the decision is taken, the investor incurs a loss in terms of switching costs that arise when personnel are trained in a particular program, the support function is trained, other programs and hardware platforms are procured and designed specially with a view to the program concerned, there are other programs that specially support the one chosen and not an alternative and so on.

Programs are in most cases adapted to the organisation and the equipment they are used in. An installed program gains the character of being highly specific to the particular application, which means that the investment is irreversible as alternative applications rarely exist. In other words the organisation incurs costs that cannot be recouped by selling off and changing software environment.

Licence terms that exclude or restrict the possibility of onward sale and therefore increase the specific nature of the investment apply to much software. When software is acquired in return for a licence payment, periods of notice and other conditions can create barriers to 'exit', that is to say terminating the obligation is subject to payment of special costs. Investors therefore decide in favour of a lock-in, i.e. tying themselves to the program supplier concerned. These form part of the value of the real option.

A customer may incur substantial exit costs if a later switch to alternative software is desired.

As both software technology development and the organisational status of the IT environment are constantly changing, it is important for an investor to be able to make choices about the time of investment by specifically weighing up the above factors in the light of their current significance.

It is essential for the investor that the time of investment is chosen so that the contribution of value in relation to switching costs is maximised. The investor in some cases can work in a lower price for a software product by citing special switching costs for example as a result of the time of investment, or as a result of the decision to change over to a different type of software.

When the decision has been taken and the investment has been made, the option is no longer valid, as a decision has been made in favour of the later switching costs. There may, however, still be an option value to the extent that there are agreements governing the possibility of exiting the technological and contractual obligations entered into with the software product. Software that follows widespread standards will have less of a binding effect on future terms of contract than unique software, just as open-source software does not have any economic ties on exit. There may, however, be service agreements and other similar agreements that tie for a period.

The consequences of the loss of option are that it is more economically advantageous for an investor to purchase subsequent versions, updates etc. as long as the purchase price of these is kept below the switching costs and any additional price for alternative software product. There need not be a *de facto* cost since only if major switching costs can be counted on with some probability will a customer defer the actual consideration of whether to switch to an alternative product. As an investor can only predict the future a short time ahead, uncertainty over the possibilities of the future will often lead to unchanged practice even where subsequent analyses may demonstrate that a switch would have been advantageous at a later time.

4.7. Upgrading in an options perspective

When updates are only bug fixes, there is an improvement in existing software. When new facilities are also added, we talk of an upgrade. Upgrading from the economic point of view is the result of an option that may have widely differing consequences depending on the context in which it occurs. In an analysis of upgrading as an option, we make clear the differences there are in switching costs between investments in proprietary software and in open-source software.

As a result of the development in licence rules it is possible – and necessary – to look at how licences can create incentives to upgrade independently of the need for upgrading that really exists. This is the case where the technological useful life and practical value of an application are sidelined by the economic terms offered by choosing a form of licence where the decision on upgrading is independent of licence payment. In cases like these, there are in other words no direct expenses in upgrading, looked at it in isolation from the follow-on effects on other associated arrangements. On the other hand, there are greater or lesser consequential costs in upgrades such as teaching costs, expenditure on related applications and replacement of hardware.

In some situations, it may be economically advantageous to take out multi-year licences with the right to regular updates of an application without utilising it, in order to *avoid* incurring consequential costs that are irreversible (cf. Table 4.1). By paying lip service to the 'rules of the game' of a licence, a licensee can maintain its freedom to update when it fits in with the company's own plans. Inadequate utilisation of access to updates presupposes freezing of other applications and also of hardware, which normally means that the customer will make use of the updates at a more appropriate time. The moment the update takes place automatically over net servers, this freedom is taken away from the licensee. In its licence rules for some software components in Windows XP, Microsoft has specified the right to automatically update and debug via the Net on the individual installation, depriving the licensee of the possibility of deciding for itself the time of updating and in so doing protecting its total installation against any compatibility requirements (cf. Microsoft's 'Product Use Rights', pages 6-7).

Upgrading is a 'limited' procurement situation, as the reason for procurement may be a desire for increased capacity, more users, improvements in the software etc. What governs the need for upgrading may be pressure coming from outside to upgrade, or it may stem from changed internal conditions. These aspects may be combined in a small number of factors that distinguish open-source software from proprietary software.

A major difference between proprietary software and open-source software is that upgrades are often tied to requirements for the operating system and hardware updating, which are difficult to evade if these ties are developed in proprietary technology. These 'ties' have significant economic consequences. Whether these continue to be created or not is a decision for the major software suppliers. The difference compared with proprietary software is that open-source software cannot create equivalent systematic technological ties as this form of tie cannot be put into effect in an open technology, where there is neither an economic incentive for this to be done nor a possibility of excluding alternative suppliers of compatible software. Including the upgrade tie in the development perspective causes the economic differences between proprietary software and open-source software to emerge more clearly.

4.7.1. Supplier earnings and upgrading

Upgrading takes place in proprietary software on the initiative of the owner of rights to the software, who has an interest in creating (greater) turnover, either by increasing the number of customers or through existing customers purchasing upgrades. The more widespread a product has become, the greater the significance of pressure to upgrade on the customer population for the future earnings of providers. It is obvious that this motive may determine the timing of upgrades, which does not necessarily coincide with the needs of users for upgrading of their software. In a mature market, where existing products meet the needs of most customers, upgrades will tend in the direction of offering facilities that are demanded by very few. By far the majority therefore end up paying a 'premium' in relation to their needs.

No equivalent pressure for upgrading comes from open-source software, as there is no

opportunity for earnings closely linked to upgrades. There is constant entitlement to upgrade for open-source software, because provision exists for this as software that is freely available. The needs of users and the convenience of upgrading rather than the supplier's opportunities to make sales will therefore dictate upgrading.

Upgrading in open-source software tends to be linked to 'freezing' of a version, so that it is easier for users to gain access to a collective (and tested) update of their software. Without these versions it might become difficult to obtain an overview of the frequency and extent of patches, because the number of 'user-developers' who contribute to improvements and new facilities can run to thousands.

Open-source software products, such as StarOffice, that are in direct competition with proprietary software will be subject to a need for updating influenced by the dominant product on the market – in this case MS Office – when major changes are made to format or functionality. Digital formats that have become 'industry standard' will thus exert pressure on all competing products to update format and/or bring about conversion technology that preserves compatibility. These 'by-products' contribute towards increasing costs in changing over to an alternative software product and thus contribute to keeping customers with the prevailing industry standard.

4.7.2. Competition pressure and updating

The other factor in upgrading is competition pressure. If several providers battle for a market, innovations may play a decisive role in gaining above-average earnings. Launching new functions and better integration across programs attracts customers from other platforms, as well as mature customers who are able to exploit the new program features. Greater customer interest is therefore achieved (or tends to be achieved) for an updated, innovative program than for a less bug-affected version of a program already known.

At the same time, innovative software products contribute to prevention of the monopolisation of software markets, as new competitive conditions (in some cases) may have the effect of eliminating the significance of a proprietary industry standard. The newly established competition only works effectively in so far as it also reduces, in relative terms, the costs of exiting and of changing existing IT environments over to the new technology. To take an example, several proprietary network protocols have been surpassed by open-source software (the TCP/IP protocols).

4.7.3. Externality costs in upgrading

A possible approach for owners of proprietary software is to fix a price and licence structure that gives existing customers an incentive to upgrade as and when the upgrades are offered, the time of upgrading being fixed in an economically advantageous manner if the supplier's choice is followed and less advantageous if updates are deferred until later. If incentives for upgrading within a particular period are created, new formats and facilities can be disseminated in such a large customer segment at a time that this can have the effect at the same time of imposing a *new requirement* with regard to compatibility in relation to other segments, which are consequently urged to update, with unhindered exchange of files and data in mind. This is, in other words, a strategy for market penetration.

A certain element of being 'out of step' in the updating of a dominant format means that those customers who change over to a new version first will exert pressure on other customers to upgrade too. Those customers who procure a new version first may be 'innovative' users or just those users who have deferred updating their software until this (for them late) time. In other words waves of upgrading are not only dependent on the customers who are most modernisation-oriented, but are an expression of the producer advantage in having non-synchronised waves of updating. If synchronisation occurred, the producer would be far more vulnerable to customer requirements for a price reduction to accept an update, and would more readily face demands for documented productivity and efficiency gains in updating. The latter gains may be particularly difficult to demonstrate to both the producer and the customer.

These waves of upgrading act as externality costs on the users with general cost-increasing effects of licence rules that are advantageous when viewed in isolation. The costs are not imposed on those involved in the investment decision, but are borne by others (third party), who are compelled to undertake follow-up conversion investments etc. In situations such as this, software upgrading does not represent a free market but a compulsion.

4.7.4. Upgrading frequencies

Surveys among Microsoft customers in the United States show the spread in upgrading frequency for server programs, operating systems for the individual PC and the Office suite (cf. Table 4.2). The pattern shows that the customers on average upgrade every three to four years, but varying over products, with the Office suite as the one with the relatively highest and server software with the lowest frequency.

There are limitations on the time of upgrading in the form of technical ties between PC operating system and Office suite. It is not possible to say how large a proportion of the upgrades are due to real needs and how large a proportion is due to the incentive structure referred to above.

Table 4.2. Breakdown of customers according to time of upgrading for proprietary software (in per cent)

Microsoft software products	**Upgrade after:**			
Product	2 yrs	3 yrs	4 yrs	5-6 yrs
Server software	13%	30%	30%	27%
PC operating system	15%	36%	27%	22%
Office	23%	28%	32%	17%

Source: ITIS/Sunbelt Software, March 2002. The population is 1500 companies from all parts of the world. Drawn up by ITIS/Sunbelt Software in cooperation with the consultancy Giga.

It is apparent from Annex 4 that there are technical ties between Microsoft products and the associated system requirements to be met by the PC hardware (i.e. the PC's processor, RAM and hard-disk capacity). This indicates that the hardware

must not be more than three to four years old if it is to be possible for the latest versions of Microsoft software to be used. The pattern for upgrading programs is therefore accompanied by updating of PC hardware. The economic burden in using software with these ties overall is therefore accompanied by costs in replacing or upgrading hardware. The system requirements for the corresponding open-source programs are substantially lower and there are therefore far fewer consequential costs for hardware (see Annex 4).

In the choice between open-source software and the dominant desktop products, the possibility of choice is accompanied by a *more expensive* option with regard to proprietary software, as the additional requirements for updating of operating systems and hardware which were apparently not part of the software purchase lead to additional expenditure in software procurement. The 'necessary' additional purchases demonstrate the value of the option in being able to choose between open-source software and proprietary software and being able to time the investment ideally to be able to skip the upgrades considered irrelevant to the organisation (unless the new compatibility requirements provide an incentive for them).

4.8. Licence ties

The difference between open-source and proprietary software was examined in Chapter 2 on the basis of the differences in copyright and licence rules, among other things. In this section, we look at what options are embedded in different licence rules. An analysis of the economic effects of the licence rules is not a substitute for legal analysis, which studies ties of a different nature.

An example of this is Microsoft's latest and current licence rules for the Windows operating system, which reflect both the use of prices to prompt purchases and the use of technological development to promote a particular rhythm of upgrading.

It may be mentioned as an example that, in comparison with the alternatives, until 1 August 2002 there was a direct economic advantage in changing to the new licence rules, because after that date it has become more expensive, according to information from Microsoft itself. In other words, Microsoft used price control to persuade its customers to change their form of licence at a particular time.

Proprietary software, particularly for products with a high market share, often has complex and extensive licence rules. The complexity arises because the customer breadth makes it necessary to have rules determined by customer type and situation. This complexity leads to a significant degree of individual judgment. For a particular customer the license terms will then be the result of negotiation and he will feel he is obtaining better terms than warranted by the rules. It is therefore difficult to avoid having a tie between customer and seller (the producer's negotiator), which is strengthened by agreements on secrecy. There is thus a classic marketing situation, where all the customers are satisfied and continue with the same supplier, because the customers each separately feel that they have struck a good deal. Open contracts would destroy this feeling for everyone but those with the best terms.

As software is dependent on other software to be able to work, making changes to software is never devoid of consequences. The advantage of one and the same supplier being responsible for the 'program pools' that have many mutual interfaces is therefore obvious in so far as this implies correspondingly fewer bugs, so that down time for the user is substantially reduced. The moment this advantage occurs in software with monopoly-like status, there is reason to expect monopoly prices, where the socio-economic advantages are counteracted by a socio-economic loss, without us being able directly to quantify this. If there are productivity gains associated with a new version or a new program, there is reason to expect that the price for it will be correspondingly higher than for previous versions or alternatives. In other words, incomplete market competition, which is the case under monopoly-like conditions, will enable the supplier to attain the value of any productivity gain for the customer by setting higher prices and not just by increasing the volume of software sold. Microsoft is attacked in several quarters for having raised prices for the Windows operating system, which can be read firstly as an effect of its monopoly position and secondly as an effect of higher productivity. It is not documented which of these effects applies (and to what extent).

The alternative to this scenario is to use software which both developers and users (customers) can correct, and which is not frequently replaced, while assuring a high degree of stability. The licence rules for open source follow the latter line of development.

For open-source software, upgrades will serve purposes more closely related to the 'user-as-developer', for which reason far more factors will apply than opportunities for earnings as a consequence of upgrading. On the other hand, it will be difficult to make the market or segments of the open-source software market 'go in step' and therefore achieve the network effects that are achieved by suppliers of proprietary software. There may be a socio-economic loss in not having someone with a 'baton' to beat the time if changes are made constantly, for example in exchange formats. On the plus side there are fewer socio-economic costs in procurement and operation than for proprietary software. With open-source software it is also possible to rally behind an open standard for exchange formats, so that basic compatibility between open-source and proprietary software can always be ensured. For this purpose a 'baton' is used that does not provide a basis for a private monopoly being established as the standard spreads.

From a supplier's point of view, open-source software will provide relatively fewer means to induce customers to upgrade than proprietary software, where the supplier can utilise combinations of innovation and the price mechanism with a market monopoly to attract customers to a new version of a program even where the customers cannot see the benefits. The incentives for open source customers are to be found in the value of bug fixes (obviously) and demonstrable effects of upgrades on functionality and facilities in *application* of the software concerned, as open source users will be inclined to consider their software in the local context, i.e. in their own IT environment. Open-source software therefore has to be accompanied by local

considerations more often than proprietary, which emerges from 'abstract' standard platforms on which the software is developed.

The competitive conditions are therefore a factor in favour of more rapid innovative development with proprietary software than with open-source software. A factor opposed to this is that a private monopoly has good reasons not to risk itself by changing its product 'too much', so that the monopolist acts in a conservative rather than innovative way. The latter factor can be observed on desktop software as well as infrastructure software, where there has been greater innovation in the small development teams than in the large established software houses. The latter have chosen to buy into the new technologies, when they have proven their strengths, but before they are mature for the mass markets, which would make them more difficult to acquire.

4.9. Economic model for an investment decision

On the basis of the preceding discussion, we can now draw up an economic model for investment in software. On the one hand we have the dividing line between the property rights perspective and the development perspective. On the other, both perspectives are included in a time progression, where the emphasis in the property rights costs is on the initial investment, while the economic factors related to the development perspective are based on a longer progression of more uncertain assessments.

An investment costing has to include both perspectives, which are contained in the factors below. This report focuses on the choice between proprietary software and open-source software. Those factors which are approximately identical in the two types of software are therefore omitted when the costing is concerned with a choice between the two types of software. In these cases, the option is only to be assessed as the difference in value between proprietary software and open-source software. If it is a matter of assessing a switch from one type of software to another, the option only has to include switching costs to the new alternative.

Table 4.3. Model for assessing the economics in open-source and proprietary software

		Difference between open-source and proprietary in a free choice situation	**Difference between open-source and proprietary in changing over from one to the other**
1	Procurement prices and/or licence costs	Measurable difference	Measurable difference
2	User-friendliness The effect of the user-friendliness of the software on indirect costs in the user environment (long 'response times', 'deeply buried' screens and functions, confusing icons or screen instructions etc.)	The working group is not aware of studies documenting differences between open-source software and proprietary software with regard to user-friendliness. We assume that these costs are a function of the specific design and independent of whether open-source software or proprietary software is concerned	
3	Requirements for education and training of end-users	The working group is not aware of studies documenting differences between open-source software and proprietary software with regard to teaching. We assume that these costs are a function of the specific design and independent of whether open-source software or proprietary software is concerned	An expense that can be estimated and that represents part of the lock-in effect
4	Requirement for learning in the internal IT maintenance function or for new service contracts with stated, chosen service aims for suppliers. See also 5 and 6	The requirement for local expertise is generally higher with open-source software than with proprietary software. At the same time, there is less familiarity with open-source software, particularly on desktop software, than with the most widespread proprietary software. It must therefore be assumed that the build-up of skills will be more expensive for open-source software	An expense that can be estimated and that represents part of the lock-in effect
5	Program-related prerequisites:		

5a	Compatibility of surrounding software and prerequisites for network interoperability and ensuring this	This cost depends on the specific software and has to be analysed in a costing. It must generally be said that open-source software is to a great extent based on open standards and must therefore be regarded a priori as being more compatible. With regard to adaptation to proprietary *de facto* standards, there is only a difference between the two types of software when a comparison is made with the software of the specific supplier
5b	Surrounding hardware: technical prerequisites for application, special capacity requirements and properties and ensuring these	Upgrading pressure (see above) will tend in the direction of proprietary software making greater demands on resources than open source
5c	Software for maintenance and support (tools, network management etc.)	There are fewer possible choices at present for maintenance and support on open-source software
6	The maintenance and skills requirements for the software and the procurement (recruitment) and operating costs of this, whether within the organisation (internal) or through service providers	The requirement for local expertise is generally higher with open-source software than with proprietary software, as the supplier usually offers training and consultancy assistance. Build-up of skills for open source is based on local initiative. The possibility of purchasing consultancy services from a third party depends for both types on whether there is sufficient market coverage for it to be a business area
7	Operational stability of the software, the supplier's bug-fixing capacity and policy (frequency of fixes, taking account of nature of bugs etc.)	Open-source software has high operational stability. For proprietary software, bug-fixing etc. is dependent on the willingness and capacity of the supplier to modify the software. Bug-fixing for open source is dependent on whether programmers who can carry out the changes are available

In the economic model for software, there are therefore human and technical 'ties' that increase costs when software and hardware are replaced. The relevance of the individual factors depends on the environment for which software is procured. If a tool for IT support without an end-user interface is concerned, there is only expenditure on the training of operating personnel. In the case of a new 'desktop', there may be expenditure on both end-users and operating personnel. The key question in an open model is whether there are economic advantages in choosing replacement or new procurement now or at a later time.

Even more essential is an option approach that includes how the technical and human 'ties' impose requirements with regard to choosing most 'appropriately' over the longer term beyond choosing the most advantageous time, as technical and human ties are important aspects in the utilisation of new technologies. Thirdly, it is economically essential that there will be debugged versions later on so that losses of up-time are minimal, without it being necessary to pay a higher price for the improved software. It is therefore necessary to set a time frame of six years, for example, so that the value of the options can be analysed.

4.10. Example of option

Let us look at an example of an option: a small pilot project can be used as an option that provides a basis for a better assessment of the value of an investment in new software *before* the investment is made. As well as being a reversible decision (it can be undone without significant losses, in contrast to the actual investment), the pilot project will also be capable of reducing the uncertainty over the nature of the advantages the investment can give the enterprise, and finally the option (the pilot project) defers the time when the decision on investment is made, if there are reasons for assuming that new, essential information will become available later, so that the likelihood of choosing the right investment is increased.

The above option has to be compared with the investment (the replacement) in either an open-source software or proprietary software solution: if open-source software is functionally identical to proprietary software in all respects (functionally equivalent) except purchase price (licences), it will be advantageous to procure open-source software at once, i.e. never to defer the decision as licence expenditure already defrayed can never be recouped and is a total loss. The size of the loss is a function of the lifetime the software is expected to have and can be calculated as the discounted value of all licence expenditure for the period. The interest can be set at the market rate for operating capital as software does not have any resale value (not permitted by the holder of the rights). In the following analyses we have not assumed full functional equivalence, but have looked at a number of types of costs that can distinguish the two basic types of software.

4.11. Conclusions

The fundamental aspect for software as a product is that the development costs are high and the distribution costs are very low. Another important aspect is that if one does not own the source code oneself, one does not buy a product but a limiting set of rights to use the product.

For standard software, the general market terms will tend in the direction of a very few suppliers or a monopoly. In situations of this kind, it will only be possible to achieve competition by taking political decisions that

assist new market participants in entering the market.

Because of investments in training, user familiarisation, interaction with administrative procedures, integration with other systems etc., software should be regarded as an option where, prior to the investment decision, an analysis is made of the long-term costs of not being able later to switch freely to other systems.

The greatest competitor in a modern market with a dominant supplier is the supplier's previous versions. Users do not have any great incentives to change over to new systems if it is difficult to document economic benefits in switching. A monopoly supplier in a mature market will therefore be forced to provide incentives that can persuade the users to change systems. Licence rules and the control of upgrading frequencies therefore play a decisive role in the economics of a dominant supplier in such a market.

The above economic model is used in the following chapters to assess the economics of open source in relation to proprietary software.

Chapter 5

Economic analyses of the use of open source on the desktop

This chapter examines the use of the standard tool on any workstation: the office suite. The choice between MS Office as the most common proprietary software and StarOffice/OpenOffice as the chosen open-source alternative is discussed on the basis of a fictitious example and three specific costings. The assessment of economics is based on the model drawn up in Chapter 4. The examples show that the economics are heavily dependent on how frequently upgrading takes place.
We have chosen to analyse the economics of the use of office suites as an example of a product that:

- Is in very widespread use - there is one or other office suite on almost every PC
- Is in a mature market, where the needs of the large majority of users have been met for several years
- Holds a near-monopoly. Microsoft Office holds a very high market share, probably more than 90% of all procurements in the area
- Can be delivered by several open source office suites, all with an extremely low market share

As explained in Chapter 3, we have chosen to focus on StarOffice/OpenOffice, with Microsoft Office as the natural choice for comparison.

This chapter analyses open-source software both from the property rights-oriented point of view of users and on the basis of a development-oriented point of view in order to assess the economic differences between open-source software and proprietary software. This is an analysis of differences between open-source software and proprietary software for one or more selected products with the assumption of 'all other things being equal'. The analysis covers the change-over or procurement situation in the narrow sense, i.e. without involving maintenance and phase-out.

Calculations on costs of open-source software in relation to proprietary software should be based on specific circumstances, but at the same time any realistic costing based on a particular situation will have many specific elements that make it impossible to generalise.

Prices and costs of this installation are obtained from suppliers and are therefore realistic and up-to-date (August 2002).

We have chosen a few entirely practical examples to illustrate the complexity of a practical analysis:

- An imaginary example that shows the economics if installation is implemented from scratch
- Århus County, which is due to change its office suite from Corel (WordPerfect etc.) to a new office suite
- Hanstholm Local Authority, which is the midst of the change-over and where a limited number of pilot users have been working with StarOffice since April 2002
- Copenhagen Employment Training Centre (AMU), which uses StarOffice in a server-based environment

Compatibility in the exchange of software is the decisive factor and is discussed in a separate section.

5.1. The general economic model

In Chapter 4 we drew up a theoretical model, which we take as our basis in assessing the costs in a comparison between StarOffice/OpenOffice and MS Office. The model is used at the start of the table to justify which costs are included in the examples and which are omitted. The examples therefore do not include all the costs in the investments concerned, as we focus on the choice between two

alternatives. We therefore concentrate on the parts of the costing in which there is a difference between StarOffice/OpenOffice and MS Office.

Table 5.1. General assessments of the difference between StarOffice/OpenOffice and MS Office (The numbers are identical to Table 4.3 in Chapter 4).

		Difference between open source and proprietary software in a situation of free choice.	**Difference in changing over from MS Office to StarOffice/OpenOffice**
1	Procurement prices and/or licence costs	The licence costs for StarOffice are modest. Regarding MS Office licences, see Annex 1	The economic costing is based on the type of existing licences (see Annex 3)
2	Effect of user-friendliness on indirect costs	This factor is not included in the costings (see Note 1)	
3	Requirements for user education and training	We assume that the costs are approximately identical (see Note 1)	Because of great similarity in the user interface, it is a limited expense (see Note 3)
4	Requirements for teaching in IT maintenance function	We assume that the build-up of skills is more expensive for StarOffice/OpenOffice (see Note 2)	Local expertise has to be built up (see Note 2)
5	Program-related prerequisites:		
5a	Compatibility and network interoperability	This cost may become quite significant (see Note 4)	
5b	Surrounding hardware: Technical prerequisites for use, special capacity requirements and features and ensuring these	A significant additional expense in frequent upgrading of MS Office (see Annex 2)	
5c	Software for maintenance and support (tools, network management etc.)	This cost is not included, because the choice of office package is not significant for this item (see Note 5)	
6	Maintenance and skills requirements of the software	This cost is assumed to be higher for open source (see Note 2)	
7	Operational stability of the software and fixing of bugs by the supplier	Costs of this are not included (see Note 1) with regard to operational stability (see Note 6), or with regard to fixing of bugs	

Notes to the table

1) The working group is not aware of studies documenting differences between StarOffice/OpenOffice and MS Office on these points.

2) The requirement for local expertise is generally higher in StarOffice/OpenOffice than MS Office. At the same time, general awareness of StarOffice/OpenOffice at present is lower than awareness of MS Office because of the modest market share. No expanded network of consultancy services is therefore available, and the range of courses available is very limited in comparison with MS Office. The PC driving licence, for example, in practice is based on MS Office.

3) There are costs for the actual teaching, but also hidden costs that are difficult to quantify for 'unlearning' old habits.

4) MS Office uses closed formats. All other producers therefore have to do 'reverse engineering', and this takes up a large amount of time and creates substantial development costs (see 5.7).

5) This is more a matter of the operating system and the overall set-up of the PC. The office suite, regardless of whether it is MS Office or StarOffice/OpenOffice, does not in itself have the great capabilities for central management, but is dependent on the capabilities of the underlying operating system for support. The tools are generally better for Microsoft Windows than for Linux, and there are therefore more third-party providers. Using StarOffice/OpenOffice on a Windows platform will provide the same opportunities for central management as for MS Office.

6) Microsoft issues frequent updates with bug fixes, but it is difficult to persuade Microsoft to include a specific, non-critical problem in its plans. For StarOffice/OpenOffice there is no experience yet of the speed of debugging based on a reasonable market share.

As can be seen from the table, there are a number of costs that are not included in the examples. At the same time, the analysis above shows that if StarOffice/OpenOffice is used on a Windows platform, the situation is substantially different with regard to administration and management than with use on a Linux platform.

5.2. A fictitious example

An idealised model installation is used to obtain a basis for assessing costs: a completely new installation without any existing stock of software or hardware.

Only costs of general tools are included in the calculations: an office suite and an e-mail/calendar

program relating to clients and the corresponding software on servers. The necessary licences are included on the clients so that the clients can be used to work with a Microsoft server, as it is unlikely that the clients will not use one or other application on one or other server with Microsoft software. The calculations are based on a supplier under the SKI agreement (see Annex 3).

The calculations do not include costs of custom built systems, nor is expenditure on general software other than that mentioned included.

We have carried out calculations for four model networks. One pair is a typical network with around 200 workstations. The second pair is a larger network with around 2 000 workstations. These two sizes have been chosen because we predict development where public decentralised networks are brought together in larger units, like the development that can be observed in larger enterprises. The concentration is justified by greater efficiency in support, operation, maintenance, investments etc. We have only shown the figures for 2 000 workstations, as there is only a modest economy of scale in this costing, with no costs that are approximately equal for the two alternatives being included. For example, expenditure on backup power supply, including wiring, will be largely identical in open-source software and proprietary software. These expenditure items are not included in the costing.

The use of server-based software for both custom built systems and office systems will probably increase during the coming years. We therefore set up the model network in two versions: a traditional set-up with clients and servers and a set-up using server-based software wherever possible, running via thin clients. The calculations for server-based software include costs for the management of clients, taking Citrix Metaframe as an example. This is strictly speaking not necessary if only the general tools mentioned are used.

All four options are provided with two sets of software:

- Open-source software to the broadest possible extent, on both clients and servers
- Microsoft Office

5.2.1. Initial costs

As completely new machines are involved, which today are supplied with Windows operating systems, costs of Windows on clients are not included.

Table 5.2. Initial costs with 2 000 workstations

	Software on the desktop (PC as clients)		**Server-based software (thin clients**)	
	Microsoft	open-source software	Micro-soft	open-source software
Per workstation - DKK	12 777	10 460	9 602	7 164
Total – DKK million	25.55	19.21	20.92	14.33

The costs are calculated on the basis of offers from a supplier under the SKI agreement

5.2.2. The development perspective

In order to find the annual costs, it is assumed that the model installation is used unchanged for a number of years. We suppose, for example, that updating of the application is only performed to the same extent as the development of office suites etc. It is obviously unrealistic for no changes to take place in the application, but we do not wish to guess at future development. We also assume that the market situation remains unchanged throughout the costing period. We anticipate, for example, that the software suppliers have an unchanged licensing policy throughout the costing period.

5.2.3. Annual expenditure on the model installation

Conversion to annual costs depends on the circumstances. The calculation below includes only expenditure on software licences and replacement of hardware, as we assume that all other costs will be at the same level for the selected alternatives.

We estimate that a well-planned upgrade will cost the same in work expended, regardless of what type of software is used, but with economies of scale in the case of a larger installation. These costs will therefore depend on the frequency of updating and not on the type of software (see Annex 3).

Two alternatives are shown for the Microsoft comparison:

- a strategy involving a strategic choice of Microsoft as a platform on the client and server and the quickest possible replacement of software on the client, i.e. every other year. This strategy requires PCs to be replaced every four years. An enterprise agreement, which is economically most advantageous for such a strategy, has been chosen here as the Microsoft licence form. This form of licence provides free entitlement to upgrade. At the same time, it will be possible, at no further cost, to update with the regular improvements which Microsoft makes available to users with this form of licence. Servers and software for servers are replaced every four years
- a strategy involving a strategic choice of Microsoft as a platform on the server, but with a change of software on the server and client every six years. This strategy requires PC to be replaced every six years. Purchase of licences without upgrading

rights in the sixth year is chosen here. This also means that there is only access to the fixes Microsoft makes available to all users
- in all the examples with thin clients, the hardware is replaced every six years.

Economics does not dictate the frequency of upgrading in the OpenOffice alternative. At the same time, OpenOffice does not impose any major system requirements, and can therefore be used on relatively old machines. Replacement of thin clients/PCs every six years is therefore included in the calculations. Regular updating is possible, but the same updating as in the Microsoft alternatives is anticipated here.

Table 5.3. Annual costs of licences and replacement of hardware for 2000 workstations

Thousand DKK	**Software on workstation (PC as clients)**			**Server-based software (thin clients)**		
	Microsoft – upgrading		Open- source software	Microsoft – upgrading		Open- source software
	Every 2 years	Every 6 years		Every 2 years	Every 6 years	
Per workstation - DKK	3 899	2 063	1 482	2 610	1 731	817
2000 workstations – thousand DKK	7 797	4 125	2 964	5 220	3 462	1 634

The costs are calculated on the basis of an offer from a supplier under the SKI agreement

This example shows a connection between upgrading of software of either the open-source or proprietary type.

As well as showing the direct difference in procurement prices, there are differences caused by derived requirements to be met in the form of hardware investments in order to be able to use the new proprietary software, while open-source software does not give rise to a requirement for new hardware at this time. The example shows that the direct difference measured on licences does not cover the total direct and indirect cost in choosing between the alternatives.

A more complete comparison is obtained by including hardware. It is not sufficient to include this, however, as proprietary software has built-in pricing for real options so that the wish to be free today or in three yeas with regard to choice of software entails greater expenditure than the direct (nominal) expenditure. The derived economic ties become more clearly apparent if the costs are looked at over a longer period of time. Varying the time at which it is desired that proprietary software will be used or ceasing to use it makes a number of costs involved in the proprietary software solution visible. The real difference between open-source software and a proprietary software solution is therefore *greater* than the direct difference in purchase and/or licence prices.

5.3. Århus County

Århus County has for many years used WordPerfect and later the whole Corel suite as the standard on desktops in its administration and in most of its education sector. This has posed an increasing problem in recent years because:

- partners with which it cooperates have increasingly adopted MS Word as their document standard
- some of the large hospitals in the County have introduced MS Office as the standard for the same reason
- MS Office is widely used on the employees' home PCs
- Upper secondary school students demand to be able to use MS Office, even if the Corel suite can be made available to them free of charge

Corel has recently decided that new versions will not be supplied in Danish in the future. A user survey has shown that many employees will not be able to cope with English texts in menus and help functions.

The County appointed a group to analyse the costs of different alternatives for changing office suite on 7 000 administrative PCs. Programs other than the office suite are only included to the extent that they are directly dependent on Corel's suite.

In this example, the costs have been modified so that they reflect the licensing costs after 1 August 2002. In addition, only the costs of replacing the office suite on the 7 000 administrative PCs are included in the calculation. No growth in population of PCs is therefore anticipated in the example.

All PCs were studied with regard to the minimum requirements to be met in order to use Microsoft Office 2000, Microsoft Office XP and StarOffice:

- all PCs used could meet the requirements for using StarOffice
- 11% could not meet the requirements for Office 2000
- 56% could not meet the requirements for Office XP

Århus County does not have any policy on replacing PCs, but on the basis of the survey and internal assessments it can be estimated that the machines have a practical life of five to six years.

The 7 000 PCs are mainly equipped with the operating system the PC was supplied with. Windows 95, Windows 98, Windows ME and Windows 2000 are used. Only a limited proportion of the PCs have Windows 2000, and even fewer have Windows XP.

The working group has carried out a costing for three alternatives:

- change-over to Microsoft Office with a licence according to the SKI agreement
- change-over to StarOffice with a one-off licence of DKK 200 to 400 per PC

- change-over to OpenOffice. There is a limited need for the extra facilities that exist in StarOffice, as OpenOffice contains a Danish spellchecker

It was decided, under all the circumstances, that the standard for the exchange of files should be Microsoft Office (see 5.7.2.).

5.3.1. Economics in Århus County's choice of office suite
The following elements are included in the economic analysis from the point of view of rights. Unless otherwise stated, these are one-off investments.

5.3.2. Licence costs
Microsoft Office: Purchase of new suites under the select agreement is anticipated, i.e. purchase of licences without an upgrading agreement.

StarOffice: It was considered possible to obtain the lowest rate to the extent indicated by Sun.

Renewal of the PC population in the purchase of Microsoft Office is at the outset cost calculated with MS Office installed on the PCs.

In the event of purchase of MS Office 2000: cost of replacement of the 11% that do not meet the system requirements for MS Office 2000 is included. These are replaced by a standard PC, including monitor, as the monitors are considered obsolete. Total DKK 6.5 million.

In the event of purchase of MS Office XP: cost of replacement of the 56% that do not meet the system requirements for MS Office XP is included. It is anticipated that half the monitors can be used. Total DKK 29.4 million.

5.3.3. Change of operating system.
It was considered that the PCs, after the replacement referred to above, did not need further upgrading of operating systems, as Windows is pre-installed on new machines.

Adjustments of graphics, templates, including tidying-up and standardisation: the work is carried out internally by staff in the IT Office, some of whom have experience with Microsoft Office.

Implementation, technical installation: experience from other counties shows expenditure of DKK 300 per PC. It is assumed that the expenditure is the same for all three alternatives and includes overheads for the build-up of skills.

Training is considered to be identical in all three alternatives. The extent of training of users and super-users has to be weighed up against the time spent by the users on building up experience. At the same time, it is expected that the training of users and super-users will to some extent be carried out by the county's own training department. Seven people in the central IT department are to go on a five-day course, 200 super-users on a two-day course and all 5 000 users are to go on a half-day course.

Project management, detailed analysis and investigation are different, as it is more difficult to purchase advisory expertise in StarOffice/OpenOffice than in Microsoft products.

DKK 1 million has been set aside as a one-off expense item for the implementation and adaptation of software to convert existing Corel documents to a format that can be read by StarOffice/OpenOffice (see 5.7.2.).

Table 5.4. Århus County: Costs in accordance with the rights perspective

DKK million (locally installed software)		**Microsoft Office 2000**	**Micro-soft Office XP**	**Star-Office**	**Open-Office**
Licences		20.1	20.1	1.5	0
Switch-ing costs	Adaptation of templates etc.	0.1	0.1	0.3	0.3
	Conversion program			1.0	1.0
	Technical installation	2.1	2.1	2.1	2.1
	Training	5.9	5.9	5.9	5.9
	Project management	0.5	0.5	0.7	0.7
Expenditure on PCs		6.5	29.4	0	0
Total		35.2	58.1	10.5	9.0
DKK per workstation		5 025	8 300	1 500	1 275

The example shows that a focus on the property rights perspective offers a great advantage for open-source software, but that the difference in the practical situation becomes greater when account is taken of the replacement of PCs. The examples shows at the same time that the costing depends on the starting situation, the situation in Århus County being that only a modest proportion of the users have a knowledge of one of the three alternatives. If one of the Microsoft Office suites had been used for a number of years, the costing would have looked different, as only StarOffice/OpenOffice would be subject to the DKK 9 million calculated here for switching costs. A switch of this kind would still be advantageous in terms of initial costs, but not by such a wide margin.

Correspondingly, the age of the existing machine population has a decisive bearing on the direct expenditure in procurement of Microsoft. It can be argued that purchases that under all circumstances would have been made over a number of years are brought forward. But as the PCs now being bought have a shorter life than the machines that alternatively would be procured in the future, the expense is real, viewed over a longer-term perspective. This is discussed later.

5.3.4. Server-based software as an alternative
Expenditure on new hardware is heavily dependent on the configuration chosen. If MS Office was to be

used as server-based software instead of decentralised installation of MS Office, the existing PCs could be used as thin clients. This would do away with the expenditure on the purchase of PCs. If replacement takes place over the next few years, instead of PCs it would be possible to purchase thin clients, which are priced at less than half the price of a PC.

On the other hand, an investment would be made in CITRIX licences, which are quite expensive and are not the subject of a volume discount. An investment would be made at the same time in substantially larger central servers which, depending on the configuration, result in additional expenditure of around DKK 1 000 per user. The ratio between the three alternatives is substantially narrowed in this costing.

Table 5.5. Århus County: Costs in changing over to server-based software

DKK million		**Microsoft Office XP**	**Star Office**	**Open Office**
CITRIX and Microsoft licences		34.1	15.5	14.0
Switching costs	Adaptation of templates etc.	0.1	0.3	0.3
	Conversion program		1.0	1.0
	Technical installation	2.6	2.6	2.6
	Training	5.9	5.9	5.9
	Project management	1.0	1.2	1.2
Extra Server Capacity		7.2	7.2	7.2
Total		50.9	33.7	32.2
DKK per workstation		7 275	4 800	4 600

In the open-source software alternatives, instead of a CITRIX licence an investment could be made in new thin clients for a total of 7 000 workstations, and the thin clients could then be managed with open-source software. This would result in additional expenditure of around DKK 10 million in the two alternatives. On the other hand, all the workstations would have new and identical equipment.

The greatest advantage in changing over to server-based software is in reduced operating expenses and better opportunities for support, backup and so on. We have chosen not to include these costs, however, as we have not, at this point, seen documented evidence of differences between open-source and proprietary software.

5.3.5. The development perspective in Århus County

In a narrow development perspective, the function of the Office suite can be viewed in relation to users' direct needs over a number of years. It is assumed in this costing that Microsoft's development and licensing policy remains unchanged over a number of years. To simplify the comparison, only the expenditure that differs for the alternatives chosen is included here. Nor are operating costs included in the calculation.

Integration of the Office suite with other systems is looked at in a broader, more realistic perspective.

5.3.6. The narrower development perspective

Two options are looked at here. In the first, Århus County considers the functionality of the office suite purchased to be satisfactory and anticipates using this unchanged over a number of years. In the second, updates every other year are anticipated. It is anticipated in the example that a number of costs are independent of the type of software chosen. The input of work in the actual upgrading can be arranged just as efficiently in both situations. Expenditure on Microsoft Back Office is not included either, as it is assumed that virtually all PCs are to have access to a server with Microsoft software. Microsoft CAL licences are therefore to be purchased for all PCs regardless of the type of software.

The cheapest form of Microsoft licence in the first option is the purchase of licence rights under a SELECT agreement, where one buys each individual licence and has to make sure for oneself that all the installations are covered by the licence. In the second, an Enterprise agreement has been chosen, where the right to regular upgrades in particular is paid for (see Annex 3).

5.3.7. Local implementation of software

If it is assumed that Århus County upgrades all 7 000 Microsoft Office suites every four years and that there is no growth in the population of machines, a full price would have to be paid for Microsoft licences on every upgrade. At the same time, both 56% of newly procured PCs (see 5.3.) and the 44% relatively new PCs would be replaced in the fourth year. If Microsoft's licence policy continues as it is at present, these will be supplied with the latest version of Windows and only Office licences will therefore be procured. 7 000 PCs and 7 000 Office licences will thus be procured every four years, giving a total expenditure of just under DKK 80 million or an average of just under DKK 20 million a year. If upgrading only takes place every six years instead, the cost becomes just over DKK 13 million a year.

If MS Office is upgraded every other year, new PCs will still have to purchased every four years. We anticipate that the operating system of the PCs will be kept unchanged for four years. A Microsoft Enterprise Agreement is paid for as an annual sum.

The annual cost in this situation is around DKK 25 million.

In an alternative with OpenOffice, licences do not have to be paid for, and at the same time OpenOffice is expected to make fewer demands on the equipment, so that the PCs have an economic life-span of six years. The comparative expenditure in this case is therefore around DKK 10 million a year.

5.3.8. Server-based software (thin clients)
If the opportunities with server-based software are looked at, the expenditure on Office licences will be the same. On the other hand, expenditure on purchasing hardware will be lower, as thin clients are substantially cheaper and have a longer useful life, set here at six years. On the other hand, an investment has to be made in more powerful servers. The expenditure on CITRIX (or similar) licences gradually disappears as hardware is changed to thin clients and the same functionality with open-source software can be achieved with thin clients. Overall, the expenditure becomes slightly lower.

Table 5.6. Costs of different alternative solutions in Århus County

	Office suite on workstation (PC as workstation)				**Office suite on server (thin clients)**			
	Microsoft			open-source software	Microsoft			open-source software
Upgrading frequency	Every 2 years	Every 4 years	Every 6 years		Every 2 years	Every 4 years	Every 6 years	
Annual cost	25m	20m	13m	10m	15.5m	10m	7m	5m
DKK annually per workstation	3 600	2 850	1 900	1 400	2 250	1 470	980	700

5.3.9. The broad development perspective
The office suite is to be included as a basic element of e-government. An office suite in particular will form part of the software that generates documents in a future electronic document and records management system (EDRMS). Århus County does not wish to develop a system of this kind on its own, and wishes instead to procure and adapt a ready-made system. There is a great risk in an EDRMS system that Office functionality will be used that does not exist in the present versions. It is therefore considered unrealistic to rely on it being possible for an office suite to be used unchanged over a prolonged period. The broad development perspective will therefore point towards acquiring the form of Microsoft licence that provides entitlement to frequent upgrading.

If Århus Amt is the only major user of StarOffice/OpenOffice, the County will have to adapt an EDRMS system to open-source software. This will entail higher one-off expenditure, which has not been estimated but will probably not amount to more than a few million kroner.

With more larger users of StarOffice/OpenOffice in the public sector, it will be possible to share this expediture.

5.4. Hanstholm Local Authority

Hanstholm Local Authority decided in the spring of 2002 to initiate a gradual change-over to StarOffice/OpenOffice on the local authority's PCs. The reason for this decision was a combination of a particularly tight economic position and Microsoft's new licence rules, which make it difficult to justify expenditure on upgrading and purchasing new licences. A strong contributory cause is the size of the local authority, as there is insufficient volume to obtain discounts for quantity. The decision brought a direct saving of DKK 300 000 in 2002. This sum would have been twice as high with the licence rules after 1 August. A decision was taken to change e-mail systems at the same time.

The local authority has around 200 PCs, of which a third are used in education and the remainder for administrative purposes. All the machines are linked to a common network with a highly standardised set-up and with the use of centralised management and administration software. There are two full-time employees to service the PCs and fifteen servers, and very little external support is used. The IT manager has the necessary skills to manage the change-over alone. The actual change-over was not judged to be particularly more demanding on manpower than implementing any other standard profile on the local authority's PCs.

The most significant programs on the administrative PCs are a screen terminal emulation for the central administrative systems, an office suite and a browser (which is increasingly used as an interface to special systems).

Apart from a special mail client, an open source version of these programs will be used. There is no economic incentive to change over to Linux on the PCs, as all the machines are relatively new and have a licence for Windows 2000.

To date, fifteen administrative users have operated StarOffice 6.0 in the beta version since 1 April. Experience has been favourable, among other things because the pioneers were selected partly according to need and partly after they had been working with complicated spreadsheets. The users have essentially accepted the new system without

any problems. As has been said, 'Microsoft isn't devoid of problems either'. The decision has the full backing of the leadership of the local authority, and the forthcoming conversion of the remainder of the PCs has been fully accepted among the users. No particularly long period of instruction has been needed to introduce StarOffice, but experience shows that the change-over does necessitate instruction. A number of meetings lasting 1-1½ hours are planned.

The IT manager anticipates that two groups of users will continue to use Microsoft Office for a time: any visually impaired users and users who have already built up advanced spreadsheets in Excel (for further details, see 5.6 on compatibility). This is not a problem, as the local authority has sufficient purchased licences for new versions of MS Office. The decision has provided freedom to set up end-of-life PCs as a service around the local authority, without this entailing extra licence expenditure or a risk of use in contravention of terms of licence. A number of PCs have been set up for example for the residents of centres for the elderly and in the refugee centre. One of the refugees is writing a user guide to OpenOffice in Farsi.

5.5. Copenhagen Labour Market Training (AMU) Centre

The Copenhagen AMU Centre in 2001 was in a procurement situation in which it had received quotations for a proprietary software system and open-source software system. In the total quotation we separate out the education part, as this report is concerned with administrative tasks. The administrative tasks are the same for the two solutions. The proposals received are directly comparable. We compare Microsoft Office with the open-source software clone StarOffice, which has largely the same functionality (about which more later).

The best of the quotations received are listed in Table 5.7 below. The table only includes the software and hardware part that *distinguish* the two solutions. Where there is full agreement between the two systems, we disregarded the component. It is a switch from Microsoft Office to StarOffice, but not much time has been spent on dedicated instruction, and the experience gained is that it is very easy to make the change. The first users attended a short course, but the assessment showed that this was unnecessary.

There are two sets of prices for the proprietary software solution. The first, and best, price is made contingent upon 62% of the administration being linked to education for which a considerable discount is available from Microsoft. The table therefore shows in the last column what the price without an education discount would be for the administrative part.

Table 5.7. The difference in price between an open-source software and a proprietary software office solution

Administration with StarOffice 5.2		**Administration with Microsoft Office**		
			Education discount	**No education discount**
Server for Gl	87 500	Server for Gl	37 500	37 500
Upgrading of server in V	62 500	MS Servers (3)	206 250	206 250
MS Office licences	22 470	MS Back Office licences	137 500	200 000
StarOffice 5.2	0	MS Office licences	193 327	328 288
		Citrix licences	150 000	150 000
Total cost	172 470	Total cost	724 577	922 038

Note on the table
Licence expenditure is stated as procurement without taking account of the lifetime during which a licence is paid for. The installation comprises 80 users. Thirty users have been able to continue using their PCs with the open-source software product, while these users would have servers in a change-over to the proprietary software system because of the greater processor power demanded by MS Office. The prices are from 2001 and are based on the two quotations. With education discounts from Microsoft, the prices of Back Office are 50% of the normal price and prices of MS Office 34% of the normal price, which applies to 62% of the licences for the systems concerned. The price of the MS Office solution is converted to normal prices for administration without an education discount in the last column. The right to make corrections is reserved as the table is not verified by the source, which is the Copenhagen AMU Centre.

The difference in procurement cost for the components concerned amounts to DKK 552 107, which means that the proprietary software solution is more than four times more expensive

than the open-source software solution. If we take the total solution where all the common elements are included, the price of the proprietary software solution becomes DKK 1 099 577 and open-source software DKK 547 470. If all the components are included, the difference between the two solutions is reduced, as the difference in licences represents a minor part of the total quotation.

If we remove the education discount from the administrative part, we come to nearly DKK 200 000 in licence fees, or a proprietary software solution that is 5.3 times more expensive than the open-source software solution for the 80 users.

We conclude that open-source software has significant advantages in a procurement context, where there is access to choosing a fully valid alternative solution from the open source environment.

5.6. Special requirements for open-source software from educational institutions

The working group has concentrated in its report on administrative institutions/workstations. Some of the assessments contained in the report can be applied to educational institutions, while others cannot.

Above all, educational institutions have a number of special characteristics in the use of IT in instruction:

- over the course of a year there are far more users than there are PCs, either because the users are on short intensive courses, or because the instruction is timetabled for a limited part of the instruction time and the PCs are shared by many students
- the individual user frequently changes PC and there is therefore a need for the user to have a profile that is independent of the equipment and for data always to be stored on the server
- particularly in primary/lower secondary school and upper secondary school education, there are many users who would like to 'fiddle with the set-up'. If the level of security is not very high, this may have the consequence that the machine is unusable and the software has to be re-installed. This firstly means support costs and secondly that some of the machines are not available for instruction.

The use of IT serves several different purposes in teaching:

- general introduction to the use of computers. OSS software can be used here in line with other software
- specific instruction in special programs
- use of general programs as tools in other teaching (word processing, spreadsheets, browsers and so on). OSS can be used here on an equal footing with other general programs
- use of specific programs as tools in other teaching (music, statistics, chemistry etc.). OSS can be used here as long as there are specific OSS tools

The choice of software is thus closely related to the purpose of the teaching, but for the most part stand-alone programs are used and there is rarely integration beyond that which exists in a traditional office suite with a database.

After the general introduction, the pupils/students should be offered a broader course in various forms of software, so that they are not tied to the software of a single producer. If this was a political requirement, the competition between the producers would be supported, as the users would find it easier to select on the basis of need and not on the basis of their unilateral experience.

It may be assumed that all the general programs are developed continuously, but at differing rates from different suppliers. Apart from the general introduction, it may be essential that the packages used are to some extent updated, particularly for directly business-oriented training courses. In the primary/lower secondary school (folkeskole), however, it is hardly essential that the general programs are the latest version.

In many teaching contexts, learning is supported by the fact that the pupils are able to use the same software on their own PCs. It is therefore an advantage if the licences permit the students to install on their own PCs as part of the educational institution's licence, or if the licence is so cheap that in practice it is no obstacle to installation at home.

5.6.1. Licences

By far the majority of software producers have special programs for educational institutions and for pupils/students, because the pupils of today are the users of tomorrow.

If Office programs are looked at, there are various solution models:

- Microsoft has special forms of licence for educational institutions with discounts of the order of 70-80% in relation to the prices that administrative users have to pay. Students can purchase, for their own PC, at the same discount as large users under volume licences
- Corel's licences are generally very cheap for the education sector and include the right to distribute free to pupils/students
- OpenOffice can be downloaded free of charge

In practice there is hardly any difference in costs for pupils/students, as private illegal copies of Microsoft Office appear to be the rule rather than the exception among pupils/students. We have therefore chosen not to include these costs in the comparison.

5.6.2. Educational institutions and thin clients

A number of schools have good experience with thin clients, combined with server-based software for users:

- operational stability is greater with server-based software
- skills can be centralised, which makes it easier to provide qualified service
- even PCs five to seven years old work extremely well as thin clients, which reduces repurchase costs and the risk of theft
- PCs as thin clients can be set up so that they are more secure than PCs with local software only

Taken together, this means that substantially more PCs are available to users at the same expense.

An argument put forward is that if OSS is used, knowledge of the software is limited among users, so that it is more difficult for them to 'fiddle around' with the set-up. This advantage will diminish if OSS becomes more widespread.

Thin clients can be set up with OSS software both as operating system and as Office system.

5.6.3. Economics in a system for educational institutions

As integration and interaction with procedures matter little to educational software, the development perspective is of very limited significance. This makes an economic calculation far simpler, as the cost of the licence is the only major factor. At the same time, there is no great economic difference between the various suppliers of general software, such as office packages.

Business-oriented training courses in particular will be obliged to have relatively up-to-date software. This applies particularly when the training course is concerned with specific software.

A comparison between Microsoft, Corel and OpenOffice shows that, depending on the type of educational institution, the costs of licences per PC vary in range of DKK 0 to 450 per PC per year.

5.7. Compatibility

Software licence savings must be compared to possible additional compatibility expenditure, incurred in order to ensure readability of data between applications without loss of information or layout changes. If there are no such changes in a document, it always has to be considered whether a solution is possible for example by using a more reliable document exchange format such as pdf (see below). In the longer term, it may be imagined that XML will be used as a document standard.
The requirement of compatibility is relevant in an administrative context, where it is common for electronic documents to be forwarded between departments, agencies and ministries. Requirements for the exchange of documents with members of the public will also necessitate using a standard format for data and files. As open document standards have not yet been implemented, administrations will be obliged to use widespread proprietary software solutions. There are two completely dominant solutions, Microsoft Word for text files or Adobe's pdf file format. The latter can be used for any office application print file.

5.7.1. Assessment of compatibility

The working group has carried out limited testing of the compatibility of StarOffice with Microsoft Office (see Annex 2). This test, together with testing in Århus and experience from Hanstholm, forms the basis for this assessment.
The degree of compatibility between MS Word and StarOffice Writer is very high. It will be possible for most documents in doc format to be opened in StarOffice/OpenOffice, edited, saved and returned to the original doc format without any difficulty. The test carried out shows that there are few problems in converting documents between the two formats. The problems that arose in conversion are predominantly concerned with layout, which in turn is predominantly based on anchoring of graphics.
The assessment reached on the basis of the test is that information is not generally lost in text documents. On the other hand, layout will be lost more often, particularly placement of graphics (anchors). Loss of information only occurred in connection with embedded objects, where these objects (e.g. an embedded spreadsheet) could not be opened.

The compatibility between MS Excel and StarOffice Calc is more problematic on a number of points. We regard the fact that references between pages are not transferred on reading into StarOffice Calc as being the greatest problem. The maximum number of rows in StarOffice Calc is 32 000, compared with 64 000 in MS Excel. Excel spreadsheets containing more than 32 000 rows will therefore not be directly converted. Experience from Hanstholm additionally shows that simple spreadsheets may result in an unusable layout, if the layout in Excel is constructed illogically/carelessly. If the layout is constructed properly in Excel, there are no problems.

5.7.2. Århus County

The working group in Århus County examined StarOffice with regard to compatibility with MS Word and MS Excel documents. Apart from problems with some graphic elements, compatibility was found to be acceptable. These problems were not considered to be an obstacle to using StarOffice/OpenOffice.
The switch from the Corel suite to a new office suite necessitates being able to read documents stored in Corel formats. Microsoft Office documents convert Corel formats without any loss of information. There will, however, be a loss of layout elements and macros. StarOffice/OpenOffice cannot convert Corel formats at all, but there is both proprietary software and open-source software that can convert the storage formats to Microsoft formats.

5.7.3. Hanstholm Local Authority

Fifteen people have been using StarOffice Beta for four to five months, and Hanstholm Local Authority has experienced few major problems in converting text from Microsoft Word to Star Writer. Problems that have arisen relate to tables with advanced layout and the placement of graphic elements, particularly in tables. Macros are not used in documents. The IT manager considers that by far the majority of users will not experience problems with anything other than graphics in 'birthday cards', and he can live with that.
In addition, the IT manager in Hanstholm expects that a few Excel users will continue to have to have Microsoft Office installed until such time as the problem with references between pages is solved. The Ministry of the Interior, for instance, supplies complex models etc. in Excel spreadsheets, and until these can be converted without any errors, MS Office will continue to be used.
The local authority does not have any experience in converting PowerPoint slides.

5.8. Market development

Open-source software at present has a large market share in server software for Internet infrastructure. This market share means that there are many suppliers of consultancy services and instruction in

open-source software in this area. Accordingly, there is some specific software based on open-source software. There are, for example, some e-commerce systems that use Apache as the server software.

On the client side, open-source software is measured in fractions of a percent.
If we suppose that Linux and StarOffice/OpenOffice on the client side gained a large market share, the economic conditions would change in a number of areas.

With the present market situation for PCs, having Linux installed on newly purchased PCs entails an extra cost, whether it is the supplier or oneself who does it. Large-scale operation makes it a cost for the suppliers to supply without Windows, but if the market share for Windows on PCs falls over the next few years, the suppliers will be able to supply PCs with open-source software operating systems pre-installed or more cheaply without operating systems.

The very low coverage for open-source software on clients also means that a market is not built up for consultancy assistance and advice on the implementation and operation of larger installations. For the time being, this knowledge has to be built up locally. If open-source software on clients gains a reasonable market share, it will be possible for skills to be purchased according to need as consultancy services. This will mean that individuals could see an advantage in specialising in open source, in a completely analogous way to becoming a certified Microsoft specialist. Some existing third-party suppliers of administrative and operating-system software would see a business opportunity in being able to handle Linux clients in the same way as Windows clients.

A reasonable market share would also mean that all sizes of public institutions could enter into a fixed-price contract with a supplier to install and run client software. In other words, a reasonable market share would normalise the market, so that uncertainty over open-source software disappeared.

5.9. Overall assessment of the examples

On the basis of the examples, the assessments are summed up as follows:

Table 5.8. Overall assessment of the examples

[The numbers and the text are identical to Table 4.3 in Chapter 4]		**Difference between open-source and proprietary in a situation of free choice**	**Difference in switching from MS Office to StarOffice/OpenOffice**
		Fictitious example, Århus County	**Hanstholm, AMU Centre**
1	Procurement prices and/or licence costs	Large differences in licence costs, including situations when Office is used on a Windows platform. Around half the total difference is made up of a difference in licence costs	
2	Effect of the user-friendliness of the software on indirect costs in the user environment	Not included	
3	Software requirements for end-user education and training	Not included in the fictitious example. It is assumed in Århus County that the costs are identical	Special training is not anticipated either in Hanstholm or at the AMU Centre. Limited time wastage therefore needs to be anticipated, which is not quantified
4	Learning requirement of the software and requirements for learning other software in the internal IT maintenance function	Not included in the fictitious example. Included as ½ man-year extra in Århus County	In both Hanstholm and the AMU Centre there were personnel with experience of UNIX etc. and the switch has been free of problems
5	Program-related prerequisites:		
5a	Compatibility of surrounding software and prerequisites for network interoperability and ensuring this	Not included in the fictitious example. A decision has been taken in Århus that the conversion is acceptable	Neither Hanstholm nor the AMU Centre regard conversion as an obstructing factor
5b	Surrounding hardware: technical prerequisites for application, special capacity requirements and properties and ensuring these	Frequency of upgrading of software is decisive for the economics of a Microsoft solution, both in the fictitious example and in Århus County	All the PCs in Hanstholm were sufficiently new, and the cost was not included in the calculation. The difference in hardware costs was substantial in the AMU Centre

5c	Software for maintenance and support (tools, network management etc.)	Not included in the fictitious example. Tools of this type are only used to a limited extent in Århus County, and implementation of open source will not obstruct this use	In Hanstholm, the switch has not signified changes in the use or set-up of management tools. At the AMU Centre, server-based software with effective administration and management of users is employed.
6	The maintenance and skills requirements for the software and the procurement and operating costs of this	There are not considered to be any differences in Århus County	Neither Hanstholm nor the AMU Centre has experienced changes in costs
7	Operational stability of the software, the supplier's bug-fixing capacity and policy		Both the AMU Centre and Hanstholm Local Authority have very favourable experience of operational stability. The implementation is less than one year old in both places, and there is no experience of bug-fixing etc.

5.10. Conclusions

As can be seen from the examples, briefly described in Table 5.8, the switch from MS Office to StarOffice/OpenOffice is associated with large savings, particularly on licences. In addition, there are savings in replacement of hardware if the alternative is frequent upgrading of Microsoft Office. With regard to other costs, experience from both Hanstholm and the AMU Centre shows that there has not been any notable difference in the switch to open source.

The examples also show that the specific installation is decisive in the assessment of economics. The calculation for Århus and Hanstholm, for example, is based on Windows continuing to be used as an operating system. No analyses of the costs in the short or long terms when an open source operating system is used on the clients have been made.

Chapter 6

Economic analyses of open source as infrastructure software

This chapter analyses open source as infrastructure software, i.e. operating system, servers etc. This type of software is implemented in very different situations, and it is therefore not possible to generalise from a few examples. At the same time, open source in the area of infrastructure is widely used, and there are several large foreign empirical studies that describe open source in relation to other platforms. We have chosen to refer to two foreign studies, conducted by recognised consultancies, and derive our conclusions from these.

Open source in recent years has been particularly strong in Internet infrastructure software. As mentioned previously, the competitors have applied widely differing strategies. A large number of software producers have included open-source products in their product range and have adapted their own applications to open source. Others have opted for a strategy of confrontation. On the basis of an economic assessment, the choice of strategy depends on the expected market value of different alternatives and a strategy of confrontation is more risky the smaller the market share is.

It is the judgement of the working group that, as infrastructure software, open source has demonstrated that it can compete on market terms and that with a sufficiently large market share there will also be enough enterprises offering knowledge and skills on a consultancy basis for open source products. It is, however, still the case that implementation of open-source software is at the user's risk, and that responsibility for bugs and deficiencies cannot be laid at the supplier's door. There is therefore still a greater requirement for

skill in being able to assess both opportunities for use and implementation and operation.

The economic model implies that there are many areas where the qualitative assessment does not *a priori* show differences between the two types of software. On the basis of the economic model in Chapter 4, we have examined the types of costs that might be relevant in a study of open source in infrastructure software. As can be seen from Table 6.1, an analysis will be very specific to the individual installation and it will therefore be difficult to generalise from a modest number of examples. A large comparative study of Danish installations is beyond the scope of this report. We have therefore chosen to report on two foreign studies and a single Danish case. Both studies are structured in a different way from the economic model, and this is commented on subsequently.

Table 6.1. Assessments of the difference between open-source software and proprietary software

		Difference between open-source and proprietary in a situation of choice.
1	Procurement prices and/or licence costs	There is a measurable difference here
2	The effect of the user-friendliness of the software on indirect costs in the user environment	As infrastructure programs are not used by end-users, this expenditure is irrelevant
3	Software requirements for end-user education and training	As above
4	Learning requirements of the software and requirements for learning other software in the internal IT maintenance function (extra courses etc.) or for new service contracts with stated, chosen service aims for suppliers. See also 5 and 6	Although the requirement for local expertise is greater with open source, learning generally takes place individually, using the Internet, in contrast to proprietary software, where supplier-specific, but user-paid courses are often held. The working group judges that with the widespread use of open source in the area of infrastructure it may be assumed that there are no significant differences between the costs of learning on the two types of software
5	Program-related prerequisites:	
5a	Compatibility of surrounding software and prerequisites for network interoperability and ensuring this	Infrastructure open-source software is in widespread use and is supported on all platforms. The economic assessment must depend on the specific circumstances
5b	Surrounding hardware: technical prerequisites for application, special capacity requirements and properties and ensuring these	Infrastructure open source is in widespread use and is supported on all platforms. The economic assessment must depend on the specific circumstances
5c	Software for maintenance and support (tools, network management etc.)	There are few options at present for maintenance and support on open-source software. The economic assessment must depend on the specific needs
6	The maintenance and skills requirements for the software and the procurement (recruitment) and operating costs of this, whether within the organisation (internal) or through service providers	The requirement for local expertise is generally higher/different with open-source software than with proprietary software. The necessary expertise for infrastructure software is on the market, and the economic assessment must therefore depend on the specific need.
7	Operational stability of the software, the supplier's bug-fixing capacity and policy (frequency of fixes, taking account of nature of bugs etc.)	Open-source software has high operational stability, and particularly on the most extensive infrastructure software, bugs are usually fixed quickly, but there is no guarantee of bugs being fixed. In the case of proprietary software, bug-fixing depends on the supplier's prioritisation of the problem

6.1. Linux vs. UNIX

In a study at IDC[37] of the economic and technical differences between Linux, which is open-source software, and UNIX, which is proprietary software, varying differences in costs were found for the TCO of the respective platforms for server programs. TCO stands for 'total cost of ownership', and includes procurement expenditure and direct working time spent on support and normalised to 1000 supported users per year, comprising support, procurement, licences and overheads.

Internet tasks comprise operation of internal and external firewalls, web services including caching, business-to-business web tasks and business-to-customer web tasks. Collaborative tasks refer to programs that support the cooperation of users in sharing information and processes and comprises common directory and messaging platforms, i.e. e-mail, common calendars, common folders and databases, threaded discussions and customised applications.

[37] The Role of Linux in Reducing the Cost of Enterprise Computing. An IDC White Paper

Figure 6.1.

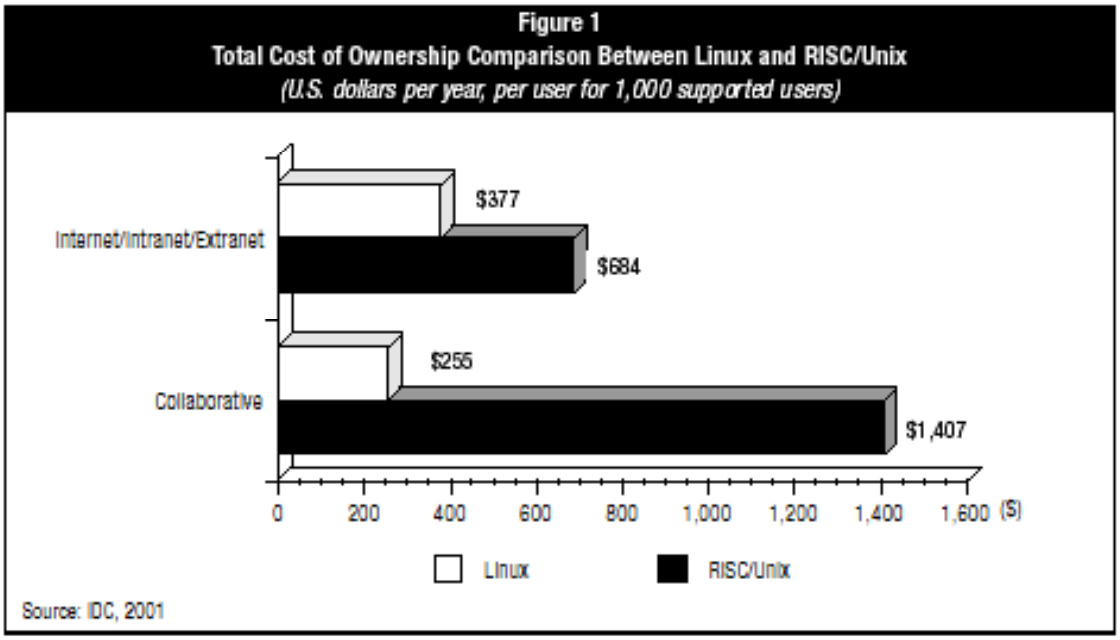


Figure 1
Total Cost of Ownership Comparison Between Linux and RISC/Unix
(U.S. dollars per year, per user for 1,000 supported users)
Internet/Intranet/Extranet
$377
$684
Collaborative
$255
$1,407
0
200
400
600
800
1,000
1,200
1,400
1,600
($)
Linux
RISC/Unix
Source: IDC, 2001

Figure 6.2.

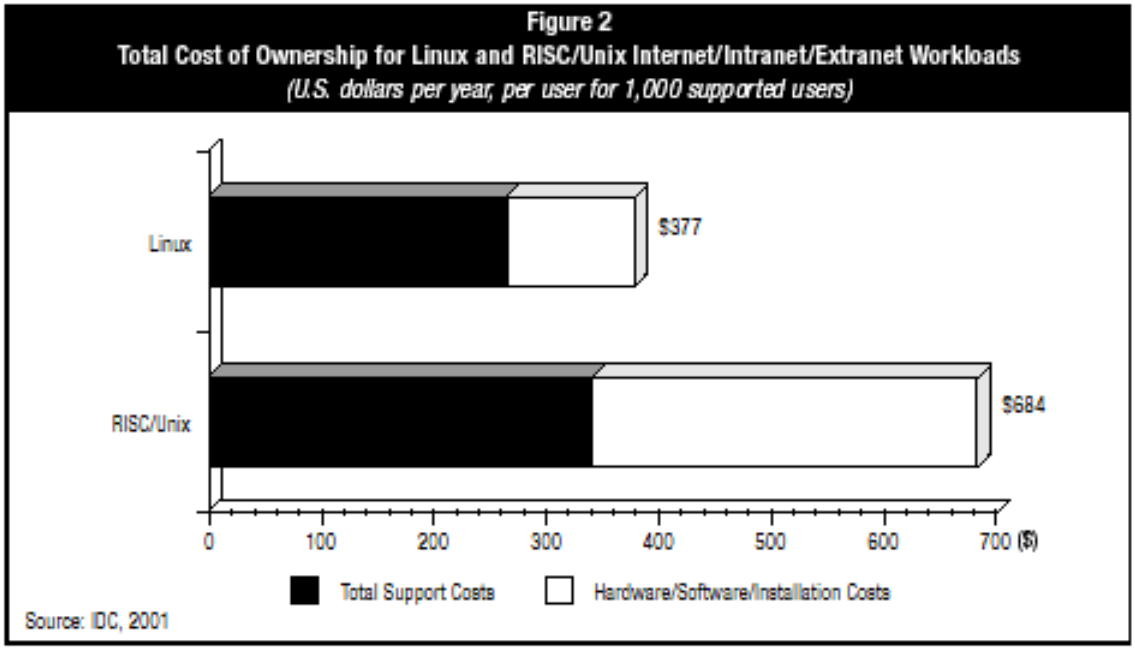


Figure 2
Total Cost of Ownership for Linux and RISC/Unix Internet/Intranet/Extranet Workloads
(U.S. dollars per year, per user for 1,000 supported users)
Linux
$377
RISC/Unix
$684
0
100
200
300
400
500
600
700 ($)
Total Support Costs
Hardware/Software/Installation Costs
Source: IDC, 2001

Table 6.1.

Table 1
Detailed Total Cost of Ownership for Linux and RISC/Unix Internet/Intranet/Extranet Workloads
(U.S. dollars per year, per user for 1,000 supported users)

Cost Area	Linux	RISC/Unix
Deinstallation and disposal of desktop systems	4	30
Procurement	7	31
Administration	27	50
Web-site management	25	40
Asset management administration	6	16
System backup	9	16
Upgrades/moves/adds/changes	11	17
Network management	26	22
Planning/management	17	12
Database management	60	54
Operations	23	13
User support	51	41
Total support costs	**266**	**341**
Per-user server hardware/software/ installation costs	111	343
Total one-year cost of ownership	**377**	**684**

Source: IDC, 2001

Figure 6.2 and Table 6.1 illustrate shifting advantages between open-source software and a proprietary software product, used as a server for Internet applications.

The open-source software product is approximately 50% cheaper overall, but with varying results in operation and administration, with some items that are higher for Linux and others that are higher for UNIX. Overall there is no great difference in day-to-day operation and administration.

The TCO for another set of applications, namely 'collaborative', is shown in Table 6.2 below. The advantages for Linux in terms of operating costs are clearly apparent here, as a proprietary UNIX is 5.5 times more expensive than Linux.

Table 6.2

Table 2
Detailed Total Cost of Ownership for Linux and RISC/Unix Collaborative Workloads
(U.S. dollars per year, per user for 1,000 supported users)

Cost Area	Linux	RISC/Unix
Deinstallation and disposal of desktop systems	8	58
Procurement	16	7
Administration	33	157
Web-site management	9	142
Asset management administration	5	26
System backup	9	65
Upgrades/moves/adds/changes	14	113
Network management	12	214
Planning/management	20	47
Database management	15	189
Operations	30	13
User support	50	145
Total support costs	**220**	**1,176**
Per-user server hardware/software/ installation costs	35	231
Total one-year cost of ownership	**255**	**1,407**

Source: IDC, 2001

The table shows that for these types of tasks Linux is less expensive than UNIX in the procurement process and in most of day-to-day operation and administration, so that Linux is particularly favourable in comparison with UNIX. In total expenditure, procurement for Internet tasks accounts for USD 31 out of USD 684 for UNIX and USD 7 out of USD 377 for Linux, equivalent to 4.5% and 1.9%. The converse situation applies to collaborative tasks, in that Linux accounts for USD 16 out of USD 255 and UNIX for USD 7 out of USD 1407, equivalent to 6.2% and 0.5%. This reflects different supplier conditions for the two products.

There are differences in the number of applications per type of server, as the UNIX servers in the reporting installations are used for more complex tasks according to the IDC study.

The difference in property rights between Linux and UNIX is not very decisive in the choice between the two platforms. Including development in the sense of maintenance, upgrades, expansions and phase-out leads to the finding that the economic differences depend more on these factors than on property rights.

6.1.1.TCO for Linux in the Enterprise

Robert Francis Group, which is an American consultancy, has analysed the costs of web servers in 2000 enterprises.[38] RFG has concentrated in particular on three architectures:

- Intel architecture with Windows and Microsoft Internet Information Server

[38] Total Cost of Ownership for Linux in the Enterprise. Robert Francis Group.

- Intel architecture with Linux and Apache web server
- Sun SPARC architecture with Solaris (UNIX) and Apache web server.

The SPARC architecture was typically vertically scaled (fewer servers with many processes each), whereas the Intel architectures were horizontally scaled (many parallel systems, each with few processors). The expenditure is therefore converted to 'normalised servers', which handle 100 000 hits a day. A comparison of this kind always raises the question whether the functions are really comparable. Although the study is concerned with web servers for all three architectures, the complexity of the individual requests differed from platform to platform. It will, for example, be possible for Linux to be distributed free of charge to many servers, and it will therefore be the preferred platform for servers that handle many simple transactions in parallel. Unfortunately, the study by RFG does not go into this.

Expenditure is divided by RFG into three principal components, all of which are assessed with a three-year time-frame:

1. Purchase of software
2. Purchase and maintenance of hardware
3. System support and administration

Other costs such as security, availability, scalability etc. were not included, partly because the participating enterprises were unable or unwilling to supply sufficient data.

Table 6.3. RFG calculates expenditure as follows per normalised web server (all figures in thousand USD)

Purchase of software	**1st year**	**2nd year**	**3rd year**	**Total**
Linux	0.4			0.4
Solaris	27.5			27.5
Windows	5.3	1.3	1.3	8.0

The expenditure reflects the pricing policy of software suppliers.

Purchase of hardware and maintenance	**1st year**	**2nd year**	**3rd year**	**Total**
Linux	37.5	0.3	0.3	38.0
Solaris	345.4	21	21	388
Windows	38.5	0.3	0.3	39.0

The hardware expenditure reflects the relatively cheap Intel processors, compared with the substantially more expensive SPARC processor. The price reflects the fact that the primary purpose of the SPARC processor is not the relatively simple handling of web queries.

System support and administration	**External assistance annually**	**Local administration annually**	**Total for 3 years**
Linux	~0	12	36.0
Solaris	19.3	29.5	146.4
Windows	1.5	46.3	143.6

RFG notes that Linux administrators would do well using more professional support, rather than relying entirely on the Internet. Solaris users made extensive use of Sun's consultancy services. Windows users typically had a support contract covering all servers.

The local administrators expenditure reflects the fact that salaries are broadly the same for Windows (USD 68 000) and Linux administrators (USD 71 000) and slightly higher for Solaris administrators (USD 86 000). On the other hand, there was a great difference in the number of 'normalised servers' a single administrator could handle: for Windows it was 10 per person and for Linux 44 servers per person. For Solaris there were just over 6, with some degree of uncertainty. Some Solaris installations, for example, had 40-60 servers per person. It is worth noting that this study shows that a system administrator can handle substantially more Linux servers than Windows servers. This calculation might suggest that the types of transactions are not entirely comparable. On the other hand, the difference is so great that there is hardly any evidence for saying that Linux requires more system administration than the two proprietary platforms. Overall, Linux was substantially cheaper to administer, despite a declared lack of administrative tools. Training, certification etc. were included in the study, but no differences were found between the two architectures, and the expenditure was therefore omitted from the comparison.

Table 6.4.

Total costs for 3 years	Total
Linux	74.5
Solaris	561.2
Windows	190.6

All in all, this report also shows that there are substantial differences in the various infrastructure architectures and that the total costs of open-source software (in this case Linux) are considerably lower than for the proprietary alternatives Windows and Solaris (Unix), as these are 2.5 times and 7.5 times more expensive respectively than open-source Linux software.

6.2. Example: Danish Consumer Information Centre

In this example, we move away from classic electronic documents to web documents that utilise the open standard HTML and the Internet, which is also based on open standards.

The application is a 'content management' (CM) system, which means that the program gives an authorised user entitlement to post content on a more closely defined part of a website. Any institution that has a website with content that has to be regularly updated by various staff can use a CM solution to its advantage, as the decentralised administration of homepages places editing of content with the person who knows the topic and not with a webmaster, who without a knowledge of the content has to edit the whole website. Content management exists in several proprietary systems (with widely differing functionality and price) and in open-source software. We have chosen to introduce this example while being fully aware that it cannot be used to generalise from.

The Danish Consumer Information Centre has chosen an open-source software solution for its content management needs. A UNIX-based system was chosen as an alternative for comparison. The two systems are comparable with regard to functionality.

Table 6.5. The difference between an open-source software and a proprietary content-management software solution

Component	open-source software system	proprietary software
Software (one-off expense)	0	260 000
Software (regular annual expense)	0	140 000
Hardware	80 000	350 000
Content management system	0	450 000
Total for procurement	80 000	1 060 000
Total regular annual expense	0	140 000

Note: Estimated prices from two different suppliers are shown for the UNIX proprietary software solution.

The Danish Consumer Information Centre's system is 'fed' into an open source environment by using the Internet and the World Wide Web for publishing. In addition, the content can be put into formats other than HTML, such as WAP and XML.

The difference between the two systems is linked to both software and computers as it is often the case that open-source software products require relatively less processor power than proprietary software products. The degree of compatibility between the two solutions is not illustrated in this example. We are not aware of the level of user satisfaction with the selected system.

Proprietary software in this example is 13 times more expensive in procurement and substantially more expensive in terms of recurring expenditure, but we do not have any information on whether the same maintenance costs will apply to these alternatives, not knowing the cost relationships that would have applied if the Danish Consumer Information Centre had chosen a different proprietary CM system as an alternative in the costing.

Table 6.6. An overview of the two studies and the example

		Difference between open-source and proprietary in the two studies
1	Procurement prices and/or licence costs	Included in both studies
2	The effect of the user-friendliness of the software on indirect costs in the user environment	Not included in any of the studies
3	Requirements for education and training of end-users	Included in IDC's analysis RFG's study relates to web servers where the user is unknown. This item is not relevant
4	Requirement for learning in the internal IT maintenance function or for new service contracts with stated, chosen service aims for suppliers	IDC: Included in analysis. RFG has included the costs in its analysis, but has found that there is no difference between the three platforms, and the costs are therefore omitted from the comparison
5	Program-related prerequisites:	
5a	Compatibility of surrounding software and prerequisites for network interoperability and ensuring this	Not included in any of the studies
5b	Surrounding hardware: technical prerequisites for application, special capacity requirements and properties and ensuring these	Not included in any of the studies
5c	Software for maintenance and support (tools, network management etc.)	Included in both studies as part of the general support, but not explicitly mentioned
6	The maintenance and skills requirements for the software and the procurement (recruitment) and operating costs of this, whether within the organisation (internal) or through service providers	Included in both studies
7	Operational stability of the software, the supplier's bug-fixing capacity and policy (frequency of fixes, taking account of nature of bugs etc.)	IDC: Not included. RFG: Discussed as a cost, but with no attempt at quantification

The two studies produce the same general results: Linux is cheaper than the proprietary operating system concerned. We have attempted below to sum up the main figures of the two studies. This summing-up is to be regarded as a rough guideline and should be viewed with caution because of differences in the type of tasks. IDC directly mentions that there is a difference, but this is not apparent from RFG's report.

Table 6.7. A summary of the two studies shows the relative costs per year

Relative figures for annual costs	**IDC – Internet / Intranet / extranet**	**IDC 'collaborative'**	**RFG – Web servers**
Linux	1.0	1.0	1.0
Windows			2.6
UNIX	1.8	5.5	7.5

As can be seen from the above table, Linux is cheaper in both studies, but with substantial differences, which may be due either to the type of task or to differences in use of the platforms.

As the reports adopt a different basis (1 000 users and 100 000 web-server hits per day respectively) and a different way of converting to annual costs, amounts cannot be directly compared across the studies. However, converted to DKK per year, the studies look as follows:

Table 6.8.

Annual costs in DKK (Rate: 750)	IDC – Internet / Intranet / extranet Per user	IDC – 'collaborative' Per user	RFG – Web servers. Per 100 000 web hits/day
Linux	2 800	1 900	185 000
Windows			475 000
UNIX	5 100	10 525	1 400 000

6.3. Conclusions

We can conclude from the two empirical studies reported on that there may be wide differences in maintenance and operating costs between proprietary software and open-source software. Both the two reports and the Danish example show that open-source infrastructure software has substantially lower costs, including unique or lower operating and administrative costs. It is worth mentioning in particular that one of the reports noted that a Linux system administrator on average handles more servers than UNIX and Windows system administrators.

Both reports compare unique types of applications, but it is impossible to say whether part of the difference is due to differences in the specific applications. On the other hand, it is not possible to establish on this basis whether these observations can be transferred to an arbitrarily chosen open-software product that can be compared with a proprietary software product.

Chapter 7

Open-source and custom built software

As mentioned previously, custom built software covers a broad range of needs, and the method of procurement reflects this range. It involves modified standard systems, function-specific systems that may have users in several institutions, and organisation-specific software that may be used in only one place. This chapter discusses first the prospects of requiring software supplied as open source and the consequences this may have, and secondly the possibility of using open source as a way of working in the development of systems, where several institutions are users with some common needs. In accordance with applicable rules, the procurement of larger systems is put out to public tender. EU Directives state that the systems have to be put out to tender every four years, but with the possibility of longer tendering periods for central systems.

7.1. Standard systems

The trend in recent years has been in the direction of procuring standard systems where possible. This form of software procurement is highly appropriate where standardised tasks are concerned, and where there are software providers in the market. It often involves modified standard systems, originally developed for the private sector. In systems of this type, the code is owned by the supplier, and the public institutions are customers in the same way as in the case of private enterprises. Adaptations generally have to be made in the same way as in the private sector, and these implementation projects may be quite extensive (e.g. DeMars, Navision Stat)[39]. There are, however, many situations where standard systems cannot be used, because the desired software is not available in the market.

7.2. Older systems and ownership of the source code

There have traditionally been two forms of ownership of custom built software:

1. Developed by and owned by a software house (e.g. financial system, owned by Kommunedata, or Sygehusløn, owned by Silkeborg Datacentral). This form of ownership has frequently arisen where there are many similar customers to share the costs of developing and operating the systems. Within national government there are, for example, some systems where the source code was originally owned by Datacentralen, but is now owned by CSC. Although the public institution does not own the source code, the right of use is normally contractually assured. Ownership is of no major practical significance for a number of large stand-alone systems, firstly because the formal owner cannot sell the system to other countries, and secondly because it will be very expensive for the public institution to replace the enterprise responsible for system development and maintenance. On the other hand, there are several examples of older stand-alone systems where the contract does not give central government the rights that would be naturally required today.

2. Developed by a software house and owned by the customer (e.g. VUE and EASY, which are owned by the Ministry of Science, Technology and Innovation and the Ministry of Education respectively). This form is particularly prevalent where there can be expected to be a single customer or a very small number of customers for a system. In these cases, the supplier cannot anticipate additional sales to cover development costs, and the customer pays all the development costs. The customer will have to pay all the costs of the system under all circumstances, and ownership can only have an effect on the timing of payment.

Ownership of the source code is essential when the systems have to be put out to tender in accordance with the EU Directive. If the supplier owns the system, there are two options: either the competitors have to supply a functionally equivalent system, or the customer has to buy the source code back from the supplier. Both are disproportionately expensive, and there is therefore no real competition for such systems. The need for change and adaptation is often met by deciding to develop a new system from scratch, entirely or partially following a tender procedure.

7.3. New development and ownership of the source code

Custom built software today will often be created as a combination of the re-use of existing modules, adaptation of these, and new development of individual modules. The supplier will re-use some code and tools from other projects. If the supplier owns the system, the customer is tied to the same supplier for customisation and enhancement. This means that other suppliers can only come into consideration if they can supply complete systems with the same functionality as those of the original supplier. Previously, when the systems were relatively small and limited, the advantages in using a single supplier and its specialists were greater than the drawback of being tied to a single supplier. Today, however, the systems have become very large and integrated as a result of many years of development. It can therefore be difficult to bring about real competition for systems of this kind. We last saw examples of this in the spring of 2002, when Kommunedata took over the local-authority systems of Columbus Data.

[39] DeMars is the implementation project for SAP in Danish Military; Navision Stat is a decentralised Navision ERP System in many state institutions.

Columbus Data did not have the resources to develop systems that could compete with Kommunedata.

If the customer owns the system, there are greater choices between potential suppliers. The situation is, however, such that the original supplier of the software, owing to its thorough knowledge of the system, has a better chance of supplying the cheapest bid for customisation and enhancement. There are, nevertheless, better ways of bringing about competition in the development and operation of such systems, and with this in mind, public purchasers should ensure ownership of the systems. On changing supplier, the customer will always be able to supply the existing source code to a new supplier. The question of open source is therefore secondary.

The drawback in requiring ownership of the source code is that the supplier will demand a higher price for the development of the system, because it has less chance of benefiting from re-using code from other projects, and because it cannot assure itself of being able to make up for lost earnings by charging a higher price for maintenance later. The higher price is therefore to some extent a matter of how payments are distributed over time.

Integration between the systems is not provided by a single supplier and has to be developed for each system. Requirements are therefore generally set for the use of standards in interfaces to other systems.

7.4. One system, several user institutions

The problems in connection with requiring that custom built software is supplied as open source are particularly relevant to systems where there are several institutions as users. The key question here is what potential and risks arise in connection with public institutions requiring that software procured by bidding/tendering is supplied as open source.

Where several institutions are to use the system, it is, in essence, a question firstly of agreeing on the requirements for the system and secondly of distributing the costs between the various institutions as users. Particularly when there are variations in needs and in time of implementation, a regulating mechanism or an institution to handle these problems has to be established. This may be a private firm, such as Silkeborg Datacentral, or a jointly owned enterprise such as Kommunedata. It may also be a form of cooperation between various IT departments. To the knowledge of the working group, it has been extremely rare to date for public organisations to have required custom built software to be supplied as open source. According to the QinetiQ report[40], open source is widely used in the hospital service in the US.

If a new system is required to be supplied as open source, this will mean that the pioneers have to pay all the development costs, because the supplier loses the option of selling licences to other customers. At the same time, it has to be possible to distinguish between the newly developed part of the software and re-used/bought-in PPS modules, which it can be particularly difficult and expensive to supply as open source. In practice, it will therefore only be possible to supply open source for those parts of the code that are not bought-in standard components.

[40] The QinetiQ report 'Analysis of the impact of Open Source Software' can be downloaded from http://www.govtalk.gov.uk/library

7.5. The need for development in e-government

e-government will require large investments in custom built software over the next few years. A large proportion of these will be made up of components that may be both standard systems and specially developed systems. These components provide the desired functionality, and many of them are standard systems that are often sold throughout the world. The components may be open source, to the extent that they provide the desired functionality. The great problem in custom built software is the integration of these components, and it is not unusual for the work on integration of components to be substantially more expensive than the purchasing of components. At the same time, control of this integration is vital to the success of the complete project. The problem for the public purchaser is to ensure sufficient competition in the supply of the software that provides the integration, at the same time as the market is very restricted, because each nation has its own structure, tradition and procedures. Although the western world is developing electronic patient records, for example, the Danish healthcare sector is only able to purchase ready-made systems in the market to a limited extent. At the same time, there will be an increasing number of components on the market as time goes by.

7.6. The European Environment Agency

The European Environment Agency (EEA) has for many years made it a requirement that custom built software is supplied as open source. The background to this is adverse experience with a joint European statistics system that is owned by the original supplier. The Agency is one of the users. It is the view of the EEA that the statistics system is too expensive to maintain and that it takes too long to persuade the supplier to make changes to the system. At the same time, however, it will cost so much to replace the whole system that the EU agrees to carry on with the existing one. This is a classic lock-in situation.

The Agency's systems are used throughout Europe, in many countries, and each individual country is responsible for their application locally. At the same time, it is the EEA that is responsible for the systems and their enhancement and integration with other systems. It is partly for this reason that open source has proved to be a great advantage. No requirements are made for a particular platform, as each individual country can adapt the software to the local platform, and each individual country can itself choose the adaptation (and the level of costs for adaptation) it wants. The adaptations made in other countries, including porting to other platforms, are also open source. This means that the EEA can concentrate on the actual functionality of

the systems and does not have to ensure usability across platforms as well. Experience with the requirement of open source has been favourable. The Agency has only made use of the possibility of changing supplier to a limited extent, partly because most applications are relatively new, but the possibility does exist. The same type of problems can be seen in Denmark, where many public institutions have the same needs for function-specific systems, but do not want to be tied to a particular supplier or platform.

7.7. Three traditional scenarios for the development of new systems

On the basis of Danish traditions in the procurement of custom built software, we have observed three types of scenarios with several public institutions as users:

1) Apparent competition – every institution has its own supplier: individual institutions, or groups of institutions, each develop their own systems with different suppliers and with the supplier owning the systems. Overall, it might appear as though there is competition, but the individual institution will be closely tied to the chosen supplier, as a switch will be associated with high costs and it will be difficult to ensure that parts of the developed systems can work together. Although the system will be subject to a tendering procedure in accordance with the EU Directive, there will not be genuine competition. 'Islands of systems with uniform functionality' will thus arise. If individual suppliers cope better in a market of this kind, oligopolies or monopolies will gradually arise. This scenario does not require any particular political backing and is the most likely if the politicians responsible do not wish to control development.

Common exchange formats: if the various institutions are to cooperate, it is necessary to create a set of standardised exchange formats so that they can all exchange the agreed data with one another. This standardisation work requires political backing for a central secretariat or something of the sort.

Example: the large complex of systems within the health service that goes under the designation of the 'electronic patient record' (EPR) is being developed in this way.

2) Self-selected monopoly: an individual supplier is given the task of developing the system, possibly with the system owned jointly by all the users. The supplier will be responsible for coordinating requirements and the pace of development and for the distribution of costs. This solution will ensure good system integration, but for large systems such as EPR and the future EDRMS it will be difficult to find a single supplier who can supply a complete system, and it will not be politically desirable to create such a monopoly for systems of that type. Although the customers jointly own the system, it will also be difficult to find real competitors when the system has to be put out to tender every four years.

Exchange: as all the users employ the same system, data can be exchanged without any problems. At the same time, it will be simpler to exchange data with other systems, as exchange formats can be agreed bilaterally.

Example: Kommunedata and the local authorities.

3) Joint specifications, which are put out to tender: a central public body is tasked with creating a uniform architecture and common specifications, the whole or limited parts of which are put out to tender. It will be difficult to carry out a task of this kind if all the users have to be involved from the outset, because all the potential users have to be considered. Protracted analytical work will be required to draw up the specifications, while the pioneers want development to proceed quickly in the areas they regard as being of particularly high priority. It is therefore more realistic for a group of pioneers to draw up the requirements, develop jointly and allow others to opt in later. This scenario requires a high degree of political control, if the final result is to be one single system. The advantages are the same as in Scenario 2, but the pilot institutions gain far more influence over the development process. A scenario of this kind requires clarification of the ownership of the system from the outset. If a decision is taken to allow the supplier to own the system, the problems in the long run are the same as in Scenario 2. If it is decided that the user institutions are to own the system, there must be political approval for an ownership structure that has to contain agreements on distribution of income between the users, both between the pioneers and in relation to later user institutions.

Exchange: as in Scenario 2 for those who choose the common system. Other users have to enter into bilateral negotiations.

Example: the Digital Task Force's tender on the future EDRMS (electronic document and records management system) is based on this type of scenario.

7.8. A fourth scenario: open source as a method of cooperation

The working group mean that a fourth scenario could be established by looking at the working methods underlying the development of a part of open-source software. We believe that such a scenario will bring major benefits in the long term. This scenario essentially signifies a shift from competition over the product to competition based on development contracts. An example that can be mentioned is the highly successful web server Apache, where a number of specialists decided to share their expertise, instead of each separately re-inventing the wheel for their own enterprises. The starting point was a web server developed for the National Center for Supercomputing Applications. This web server had been copied and modified in a number of enterprises and institutions, and an 'Apache Core Group' was formed to guide development, with just over 30 members, who take decisions on development by voting. The actual development is undertaken by a far larger group of developers, who pass suggestions to the core group. The suggestions for the most part come from active users of the web server, who have a special need. It may also be software companies that decide to use

Apache, as when IBM decided in 1998 to port Apache to all AS400 platforms.[41] The advantage in this scenario is that whenever a modification is made, the controlling group can choose between different suggestions and that those who make the suggestions can inspire one another. There is thus healthy competition in the work.

A similar working method could be used, for example, in connection with bespoke systems in Denmark while acknowledging that it will ultimately be the taxpayers who pay for the system. It is unrealistic to believe that programmers throughout the world, or even in Denmark, will queue up to develop and improve a bespoke Danish system for free. The working method therefore has to be adapted so that all software is developed under contract.

The working group's fourth scenario entails a single institution, or a group of public institutions acting jointly, drawing up requirements for a new system. These requirements are specified in tenders, in their entirety or in limited parts with the stipulation that newly developed software *has to be open source* and that bought-in components have to be well-defined and with a *de facto* standard interface. Those who take the initiative will have to pay for all the development costs in this situation. The benefits come in the slightly longer term, when institutions other than those that take the initiative have to use the system. The condition to be met for them to be able to use it must be that any modifications, additions and expansions are paid for by the institutions that need them and that it is made available as open source. A fairly large library of modules for such a system will thus be developed over a number of years. At the same time, it will be possible to obtain real competition on tenders in accordance with the EU Directive, as several enterprises will have worked with the system and be able to offer enhancement.

This scenario requires that a central controlling secretariat to coordinate the total system and its development is established no later than at the time of the first implementation. This may either be a public organisational unit, or the work can be outsourced to a private enterprise. The essential point is that a coordinating office of this kind must have significant technical and political skills. The work will be more extensive than fixing exchange formats in Scenario 1, but less than the extensive work in Scenario 3.

The task of the secretariat will be to:

- control versions of both the total system and individual components
- decide whether modifications to the core of the system, to existing modules or interfaces between modules are to be included in a new version or not
- decide whether new modules are to be part of the total system, or whether the module is entirely the responsibility of the developing institution
- ensure that all modifications and new modules fulfil both standards and quality requirements

The open situation may mean that some modules exist in competing versions, but the requirement of open source will make it possible for the institutions to choose between the different versions. The total system will therefore have many facets.

7.8.1. Political prerequisites for the fourth scenario

The establishment and operation of a controlling body of this kind and the initiation of the pilot project require political backing, but at the same time the project is risky from a political point of view. There will not be clear project management for the complete system, because the continued progression will to a great extent be based on the initiatives of individual institutions. If a single sub-project with a modification or addition is a fiasco, it will only affect the institution that initiated the sub-project. No others will start using a module that does not accomplish its task satisfactorily.

It will be difficult to assign responsibility for bugs and defects in the total system in this scenario. This will therefore make great demands on the technological skills of the institutions, and will raise the quality of the work on the system, as the customer is unable to take cover behind the supplier's skill. The scenario will therefore be an incentive for efficient project management. At the same time, the applicable system of approval is an obstacle to putting the scenario into effect. Firstly, the scenario requires the expenditure profile to be shifted so that all the development costs are paid at the start. This will mean that, at the time of the decision, the system appears to be substantially more expensive than the alternatives. On the other hand, it will be possible for later development to take place in a competitive market, where several suppliers can bid on equal terms. Secondly, the pilot institutions have to accept that they alone bear the initial costs. All in all, it's clear that the fourth scenario means not to follow the 'path of least risk' in choosing a more traditional solution, but rather to be willing to take the necessary decisions.

The advantages with the method in this scenario are the same as if one supplier were allowed to supply the system. There will only be one system in Denmark, and it will have one common interface towards the administrative user, towards the ordinary user and towards other systems. Integration of the information of different Danish authorities, stored in the common system, will be simpler. Interaction between the users will be simpler to control, and staff will only have to acquaint themselves with one system. Combining different authorities, for example mergers of local authorities or moving fields of responsibility from one ministry to another, will be simpler.

In contrast to Scenario 1, different larger and smaller software suppliers will be able to contribute to the system in maintenance and new development. It can be said that the competition is moved from the product to the producers, as every tendering situation will be

[41] Joseph Feller & Brian Fitzgerald, 'Understanding Open Source Software Development', Addison Wesley, 2002

open to all. There will also be a sufficient number of consultancies, which can see an advantage in offering skills in operation and use of the system.

7.9. Conclusions

It will not be relevant to recommend open source, as we see it in desktop and infrastructure software, in Danish bespoke systems. No one will work for free for such projects, and any development will therefore necessitate meeting the costs of development.

Proprietary systems entail a close tie to a single supplier, and in reality this eliminates competition, so that EU tendering rules have no practical effect. Non-proprietary systems are more expensive in actual development, but provide an opportunity for greater competition in continued development and are therefore cheaper in the long run. With more users, ownership requires a political structure for decisions and for the distribution of costs.

An alternative is to use open source as a working method, and in so doing to bring about greater competition on the development of systems, in whole or in part. This requires the political will to take the necessary decisions.

Chapter 8

The socio-economic consequences of open-source software

Open-source software has been the subject of a widespread myth, to the effect that removing the licence fee – which only accounts for a small part of total IT costs – does not make up for significant uncertainty on quality, performance and maintenance costs in open-source compared with proprietary software. The examples in the previous chapters will lay this myth to rest. All that remains then is to quantify the possible differences between open-source and proprietary software using a socio-economic yardstick.

The purpose of this chapter is to illustrate the socio-economic differences between the use of open-source software and proprietary software in public administration in Denmark. A socio-economic analysis assesses the total loss that follows from decisions taken against the background of limited information and imperfect market competition. It should be emphasised at the outset that there is no information as yet, in available statistics, on even the most basic use of IT in public administration. We consequently do not have any information on how widespread open-source software is in public administration in Denmark, but our general impression is that it is used to a marginal extent. The following calculations are therefore based on rough estimates and specifically reasoned assessments in each individual case. Careful estimates have been used as a rule, so that all the calculations are intended to show possible socio-economic gains or losses under well-defined conditions.

In order to be able to assess the socio-economic consequences of replacing proprietary software with open-source software, a comparative study is made of the particular types of software, using relevant assumptions from the general model (from Chapter 4) for the types of software concerned. This chapter is principally concerned with infrastructure and desktop software, because there is a possibility here of extrapolating from previous studies.

Calculations of the socio-economic consequences are based on foreign analytical results, *assuming* that they can also be generalised to the public sector in Denmark, Danish conditions being involved in the calculations as far as possible. This is a debatable assumption, but the best that can be done in the circumstances, using cost differences from the US, which for many reasons has software markets more open to competition than in Denmark. These cost differences will consequently under-value rather than exaggerate the relative differences between open-source and proprietary software, on the basis of the observation that stronger market competition leads to lower (actual) prices. The following calculations are therefore to be regarded as *rough estimates,* which indicate some relative size ratios of a more closely specified nature. We consider that the advantages in stating rough estimates of size ratios outweigh the drawbacks that follow from the uncertainty over the possible socio-economic gains, uncertainty that will always be associated with calculations of this kind, but which should not prevent efforts to form a picture of the economic proportions. These calculations should be included in the strategic considerations that become necessary when considering the volume of the total public investment in IT.

8.1. Qualitative socio-economic assessments

In Chapter 4 we put forward a general model for investments in open-source software (Table 4.3) in different situations of choice. In this chapter we study the socio-economic differences in choosing between open-source and proprietary software in a situation where no investments have yet been made and in a situation where investments have been made in proprietary software, but upgrading has become possible (for example from Windows 97 to Windows XP). We discuss the latter as a change of software platform, where replacement of the platform used hitherto can take place either by choosing new proprietary software (possibly implying change of hardware), or by choosing open source (possibly keeping existing hardware). In Table 8.1 we make the results from the studies in Chapters 5, 6, 7 the basis for the estimates and assessments shown, where these are relevant.

Table 8.1. Cost components in new procurement and changing of software

	IT cost components	**A: Difference between open-source and proprietary in a situation of choice *without* prior relevant IT investments**	**B: Choice between open-source and proprietary software in a *switch from* proprietary software**
1	*Price and/or licence fee* Procurement and/or licence costs.	Measurable difference	Measurable difference
2	*User-friendliness* The effect of the user-friendliness of the software on indirect costs in the user environment (long 'response times', 'deeply buried' screens and functions, confusing icons or screen instructions etc.)	The working group is not aware of studies that provide evidence of major differences between open-source software and proprietary software with regard to user-friendliness. We assume that these costs are a function of the specific design and independent of whether open-source or proprietary software is concerned	
3	*End-user training* Requirements for education and training of end-users	The working group is not aware of studies providing evidence of differences between open-source and proprietary software with regard to learning. We assume that these costs are a function of the specific design and independent of whether open-source or proprietary software is concerned	Where there is no functionally equivalent open-source alternative to proprietary software, there is additional expenditure on training. One course day including course payment represents a value of 1% of the annual norm of salary (cf. Table 8.4)
4	*Training of IT staff* Requirement for learning in the internal IT maintenance function or for new service contracts with stated, chosen level of service agreement for suppliers (see also 5 and 6)	The requirement for local expertise is generally higher in the case of open-source software than with proprietary software. At the same time, familiarity with open-source software, particularly for desktop software, is less than with the most widely used proprietary software. It is assumed that the build-up of skills will be greater for open-source software than for proprietary software, which puts the emphasis on the supplier retaining control over its software, in contrast to open source	Switch to open source will normally be accompanied by requirements for courses, the extent of which will depend on prerequisites for skills. A switch to a new, upgraded version of proprietary software is generally accompanied by a need for continuing training (certified supplier-specific courses). Difference in scope and price of continuing training rules out a clear conclusion in choosing between open-source and proprietary software. Both cases will imply using several per cent of the annual norm for salary costs in the own IT department or for hiring external consultants.
5	*Program-related prerequisites:*		
5 a	*Software compatibility* Compatibility of surrounding software and prerequisites for network interoperability and ensuring this	Infrastructure and desktop open-source software can be used on almost all platforms. Desktop open source has a smaller installed base, so that integration with the surrounding environment has been less thoroughly tested. For open source there is in principle better opportunity to integrate with surrounding software if this is also open-source, while the program interfaces of the proprietary software may limit the scope for integration	
5 b	*Hardware prerequisites* Surrounding hardware: technical prerequisites for application, special capacity requirements and properties and ensuring these	Infrastructure open-source software is widely used internationally and supported on all platforms. Desktop open-source software does not impose great hardware requirements	Newly upgraded version of proprietary software has normally required hardware with faster microprocessor and larger working memory. There is additional expenditure on hardware with proprietary software in comparison with open source
5 c	*Software tools* Software for maintenance and support (tools, network management etc.)	There are maintenance and support tools for proprietary and for open-source software, particularly for infrastructure and desktop software. There is additional expenditure for	Additional expenditure on software tools for proprietary software may possibly be limited by updating tool used to date instead of buying new one. There are generally competing

		proprietary software as the equivalent to open source is freely available but not always in 'ready-made' suites	maintenance tools for the most widely used infrastructure products. There is no expenditure for open-source tools
6	*Skills requirements for software tools* The maintenance and skills requirements for the software and recruitment and operating costs, whether within the organisation (internal) or through service providers	The requirement for local expertise is generally higher in the case of open-source software than proprietary software. At the same time, familiarity with open-source software, particularly desktop software, is less than with the most widely used proprietary software. It must therefore be assumed that in the present situation the build-up of skills will initially be more expensive for open-source software but will be applicable for a longer period	Proprietary software is known to IT staff. There is, however, normally a requirement for certification in newly upgraded software (continuing training requirement) created by the supplier. Learnt open source is applicable over a longer period, as there is no unavailable code but free entitlement to work with the software. There is normally less discontinuity in upgrades in open source – i.e. assurance of compatibility. It is assumed that the build-up of skills will be greater than for proprietary, which emphasises the supplier retaining control over its software
7	*Software stability* Operational stability of the software, the supplier's bug-fixing capacity and policy (frequency of fixes, taking account of nature of bugs etc.)	Open-source software has high operational stability and bugs are fixed quickly, particularly on the most widespread infrastructure	Operation is adapted to the up-time requirements of the organisation, with a justified expectation of improvement in this on changing over to a new upgrade (and measurements of this show that this is the case to a varying degree). On changing over to open source, the level of up-time is not known in advance. Documentary evidence of improvement with open source must await testing (and measurements show great stability of infrastructure and desktop open-source software)

In addition to the direct cost elements mentioned above, we have identified in Chapter 4 irreversible cost factors in software investments related to differences between open-source and proprietary software (see Table 4.1). The model is used below to illustrate socio-economic consequences of the choice between these types of software. In contrast to Table 4.1, we discuss in Table 8.2 the socio-economic effects of irreversibility, these corresponding to the additional expenditure that a switch to an open-source platforms from a purely proprietary environment entails.

When we speak of irreversibility, it also reflects the fact that previous investments, for example in skill-developing certification courses, lose their value (or most of it) on the change-over to another platform, when skills are relatively strongly supplier-specific with a limited possibility of re-use. Requiring *both* new investment in skills and the writing-off of investments made to date in building up skills makes chosen proprietary software a stronger tie for cost reasons than that related to the advantage of reduced licence cost.

Irreversibility also follows with a shift in basic formats (e.g. file formats for office software), where a dominant supplier attempts to move its market to a new 'standard' to weaken new and smaller competitors. Conversion is made more difficult until news tools are developed, after which conversion expenditure makes a further contribution to making it difficult for an alternative platform to be maintained. Control of widespread formats, for example, is used as a strong competition-guiding instrument, albeit always with risks of negative customer response, since any productivity benefits to the customers with the new format are accompanied by an increased tie to the supplier. As only the supplier of a dominant software product can implement changes of format, the supplier concerned decides what changes take place and when. These decisions are taken on the basis of the supplier's assessment of the competitive terms.

As we have argued previously, competitive conditions are a function of the software decisions the market takes. If the market takes a decision on a short-term basis, it is without an assessment of the circumstances mentioned here. If an intermediate perspective is applied, the assessment of software investment may include said circumstances. If the market is characterised by many small decision-makers, it is difficult to escape the advantages of doing the same as everyone else (accepting dominance), while the decision of a larger group on a collective switch will quickly have a visible effect on the market, as the dominant supplier will be

compelled to respond.

Table 8.2. Factors for assessing irreversibility of investment in software products

Option feature of software products	Definition	Option value	Open-source vs. proprietary software
Specificity	Any software imposes a set of requirements for its HW and software environment	Choosing software entails a degree of lock-in, i.e. incontrovertible loss on switching to alternatives which is greater the more specific software is in its requirements for its environment	Proprietary software by nature is likely to increase the requirements for the use of specific software from the same supplier. Open source allows increased investments in software, as this software can normally be used in many environments (this may possibly not be applicable to custom built software). Equivalent broad use is aimed for with international standards for proprietary software
User learning curve	software is knowledge-based product with a learning time for users	Loss in training costs is accompanied by indirect losses in reduced production during training time compared with the case of full experience	Open source has less discontinuity for users than proprietary, where there is an incentive to make software 'obsolete' to increase the market
Compatibility	How capable an software product is of working together with other software	Complementarity effect (positive economic value) and the opposite (if there is a requirement for conversion routines etc.)	Proprietary software is normally backward-compatible but rarely forward-compatible precisely so that a new market is created (see above). Proprietary software is more strongly integration-oriented in order to increase the market. But at the same time problems are created for third parties. The lock-in effect is increased.
Support learning curve (maintenance and prerequisites for support)	Skills of IT department in supporting users and maintaining software	Specific investment in staff (learning curve) and software tools, which cannot be fully re-used, entails loss on switching software. The costs in switching are a reason why suppliers of dominant software can charge a premium price compared with the smaller suppliers.	Proprietary software is dependent on courses that give precisely the insight to administer their product. Open source does not have any precise demarcation of what is 'sufficient' as it depends on the level of aspiration among the operating personnel of the organisation. Alternatively consultants may be used.
Integrability	Any software environment needs to effectively integrate new software, and software products can do this more or less effectively	Software integration costs increase in line with the incompatibility of software products and increase the barriers to acquisition of new software. Open standards reduce the barriers (all other things being equal)	Proprietary software defines interface on the basis of competitive terms, open source defines them on software technology grounds

8.2. Quantitative socio-economic assessments

Quantitative models make great demands in relation to data collection, as is the case with the models used by consultancies in comparisons between alternative products, as has become apparent in previous chapters. The working group has not gathered data from all public authorities, or even a selection of them, to illustrate socio-economic alternatives. The best basis for comparison is to find comparable installations of the alternative technology, which we have documented in individual cases in Chapters 5-7.

We show that the difference between open-source software and proprietary software is not limited to the licence price. Operating efficiency in the infrastructure examples we have shown in Chapter 6 showed surprisingly great advantages for open-source software, where relatively higher maintenance and support costs might have been expected. An American consultant report (Robert Francis, 2002) shows that Windows server software (using IIS as web server) requires significantly more maintenance than both Linux and UNIX server software (which both use open-source Apache web servers). The report finds, for an approximately standardised analysis of 'total cost of ownership' in a selection of 14 enterprises, that UNIX (Solaris) is 7.5 times and Windows 2.5 times more expensive than Linux over a three-year period. The study has not been able to quantify breakdown as a result of virus attack or rebooting of Windows servers following the installation of new 'patches',

although this is a relatively common maintenance task that distinguishes Linux and UNIX from Windows, in that they do not require rebooting. We take as our basis below a study that finds that TCO differences between Linux and UNIX of the order of 1.8 to 5.5 for UNIX depending on the set of tasks.

A Gartner report ('How to Avoid Pitfalls and Save Money With Linux Servers', research note 19 June 2002) emphasises the importance of building up local skills in IT support and the need for consultancy agreements for open-source products as well as differences in the nature of tasks in the assessment of the overall economic advantages between proprietary UNIX, Windows and open source, which confirms the relevance of our economic model (Tables 4.3 and 8.1). On the other hand, neither of the reports attempts to make a complete assessment of the significance of irreversibility other than highlighting some of these factors as a cause of a relatively smaller advantage in switching to open-source infrastructure software.

It must be emphasised that the chosen examples do not cover absolutely any open-source or proprietary software. We have chosen to base our analyses on software in very widespread use. We cannot claim on the existing basis that open-source software will *always* be more advantageous than proprietary software, firstly because there is no documentary evidence that design methodology and support for open-source software *necessarily* provide better-quality software (containing fewer bugs, greater user-friendliness, easier integration etc.), and secondly because there is no evidence that it results in quicker software development than other methodologies.

The studies undertaken in this report provide evidence for the chosen examples that substantial economic advantages have been observed for open source in *desktop software* (office programs and operating systems) and in *server operating systems*. We have presented this result while being fully aware that we do not know of all the conditions needed for us to be able to generalise. This reservation is mentioned because reports from suppliers of software are subject to the same (or equivalent) problems in establishing all the conditions that need to be met for it to be possible to generalise from study results. We have refrained from assessing the productivity differences among software, as these assessments are particularly sensitive to local organisational circumstances.

It has not been possible to calculate what proportion of the total software expenditure the selected type of software represents in the public sector in Denmark. The selected software products are characterised by a very large number of users, so that this software has a strong bearing on the socio-economic calculations, while custom built software will normally be in substantially less widespread use, that is to say the number of users will not have such a strong bearing on the calculations, whereas other factors play a more essential role.

We have not included the value of smaller software imports in our socio-economic assessments of open-source and proprietary software. In our analyses, proprietary software is an imported product, while open source can be imported without payment of licence fees. The import value is estimated to be less than the total licence expenditure.

We have not been able to identify greater intensity of work in open-source software maintenance and development than in proprietary software on the basis of the available data, and we therefore do not anticipate a net effect on employment from a switch to open source. On the other hand, we expect the skills profile of IT staff in an open-source environment to differ from Microsoft user environments, because Microsoft to a great extent uses certification courses for each individual product, while open source uses general skills, as well as access to 'Net-based communities' as support for the individual software products. We have not attempted to calculate the costs of changing skills profile. It emerges in one of the studies in Chapter 6 that IT administrative salaries in the US were marginally higher for Linux (open source) than Windows, but substantially lower than for UNIX/Solaris. It emerged at the same time, however, that Linux personnel administered far more severs per person than Windows, so that efficiency with Linux was substantially above the other platforms.

How large a share of public-sector expenditure on IT is accounted for by the selected products in desktop, infrastructure and custom built software? The working group has analysed this question in the light of the lack of statistical knowledge on IT operating expenditure broken down into specific types in the public sector and has come to the conclusion that desktop and servers probably constitute an ordinary administrative workstation, but that special software, both collective and split between workstations, cannot be assessed. Consequently neither can we determine the proportion of the former products in total IT operating costs. It should also be mentioned that the estimates presented here will be affected by any change in competitive situation between the chosen types of software, so that estimates cannot be extended an arbitrarily chosen time into the future.

When we have needed to find factors for scaling up from the individual examples, we have used some factors as keys to the socio-economic impact assessments. Scaling factors put the economic differences found in the use of IT by the Danish public sector in a social perspective. To reduce distortion in the calculations, we assume there is a 'functional equivalence' so that the alternative software provides the same service to the users. Our scaling-up constitutes an estimate which cannot meaningfully be based upon too detailed assumptions as these will not scale to society level. We reduce the number of factors substantially through this procedure.

We have not been able to calculate compatibility and switching costs on a transition from proprietary software to open source, although these would be of interest in illustrating costs in any selection of an open-source strategy (more about this in Chapter 9). We have, however, attempted to assess what a change in desktop software means for the end-user and the training expenditure on support staff, but have used this solely to assess the economic significance of differences in the frequency of upgrading software over a number of years. The scaling factors used are listed in Tables 8.3 and 8.4, which show the basis of calculation for the socio-economic assessments in Tables 8.5 and 8.6. Brief reasons for the individual scaling factors in Table 8.3 are presented here.

8.2.1. Explanation of scaling factors

The total IT equipment of the public sector is not computed. To be able to assess the potential for switching to (increased use of) open-source software, it is necessary to estimate the number of PC users. Three sets of statistics are taken as a basis for this. In Statistics Denmark's specification of the number of wage-earners for 2001, there are 823 586 full-time and 110 164 part-time employees divided between companies, local authorities and county authorities (social insurance funds and other funds, accounting for 2 281 wage-earners, are not included). Statistics Denmark's study of 'Use by families of the Internet, 4th Quarter 2001' shows that 94% of all white-collar staff have access to the Internet, and of these 11% from home only, which means that 83% of all white-collar staff have access to the Internet at work, which means the same as having access to an IT workstation. This includes both publicly and privately employed white-collar staff. As there are a total of 933 650 publicly employed wage-earners, assuming that there is no essential difference between the IT access of privately and publicly employed officials, we might expect that the maximum number of users must be of the order of 83% of 933 650, approximately 775 000. If we deduct all part-time employees, we instead reach a figure of 680 000 employees. We will correct this figure for differences associated with the training of personnel.

The above figures may include groups of officials with substantially less access to an IT workstation, as many positions such as those of carers and educators do not provide access to an individual workstation. The number of persons in the group of public employees whose highest level of education is 'vocational' is 226 964 according to Statistics Denmark's survey of the highest level of education by areas etc. in 2001. The number who have followed further education courses of medium length, such as teachers, nurses and child and youth workers is around 227 000.

Statistics Denmark's study no. 26, dated 25 January 2002, shows that 30% of workers had access to the Internet from work. We can use this low level of access for white-collar staff with medium levels of education holding limited administrative positions. We have chosen the lowest IT access factor in full knowledge that virtually all office staff have access to IT at public workstations. On the other hand, the factor is probably on the high side for many carers and teachers. Part-time employees will often share access to IT, and we therefore build in a correction for the fact that part-time employees account for 11.8% of all public employees. The correction reduces the calculated IT access to 0.88, where part-time employees are completely deducted from figures for access to PCs. Although part-time employment is highly correlated with level of education, we have applied the correction for the whole of the public sector and not just for individual groups.

Table 8.3. Calculations of the number of PCs in the public sector in Denmark

Wage-earners in the public sector in 2001		**Access to Internet (at work) for both private and public employees**		**Highest level of education for publicly employed wage-earners**			**PCs in the public sector**
Full-time	Part-time	White-collar	Blue-collar	Vocational	Further educ. of medium length	Rem-ainder	
823 586	110 164	83%	30%	226 964	227 592	479 094	
PC frequency				0.3	0.3	0.83	
Access to IT at work				68 089	68 277	397 648	
PC workstations		Correction for part-time employees					469 932

Source: Statistics Denmark, Employment statistics for 2001. Highest level of education classified by area etc. 2001, and News from Statistics Denmark, Use of the Internet by families, 4th quarter 2001.

The estimate of PC workstations in the public sector is rounded to 450 000.

We assume that each change of desktop (for both proprietary and open-source software) entails a loss of productivity comprising time spent on training instead of work and a period of relatively lower productivity, which corresponds to individual members of staff having a period of learning until they are fully conversant with the new software. We have estimated that this period of learning on average is equivalent to one working day per year, which we have converted to a value target equivalent to 1660/1924 x 7 x standard annual pay ≈ DKK 1 000, with the additional assumption that gross average pay for an IT workstation user is DKK 300 000 per annum.

A standard amount for a year of support is calculated from frequently published annual expenditure distributed between PC workstations. The standard amount for a year of support is based on an estimate of DKK 2 000 for time off work and DKK 8 000 in course costs, which is equivalent to DKK 100 per user per year.

The chosen comparisons are ones that typically cover a public IT workstation, where a PC is equipped with a network card, operating system

and office program and has access to large public systems (civil registry number system, Central Customs and Tax Administration, KMD systems etc.). The possibility of using 'thin clients' may defer the replacement of hardware, as only screens are sent to the individual PC, which is linked to a server on which the operating system and office program run. We do not have any systematic examples of this solution being applied in our material, and we have therefore omitted this alternative from our calculations.

Table 8.4. Scaling factors for the total public sector in Denmark

Scaling factor	Index	Comment
Number of computers in the public sector in Denmark	450 000	Stated separately cf. above
Number of users in the public sector in Denmark	450 000	This number is probably higher than the number of computers due to part-time work and joint use in many educational and treatment institutions. We have chosen a low estimate.
Number of servers	Figures for this are laid down in the example calculation.	There are no available statements of the number of servers (and their configuration) for the public sector. Statements of the economic differences between different software installations per user are used as an indicator
Server operating system life	An estimate of life is taken as a basis: 5 years	American data show approximately this life
Desktop operating system life	An estimate of life is taken as a basis: 4 years	American data show approximately this life
Desktop software: Office suite life	An estimate of life is taken as a basis: 4 years	American data show approximately this life
Hardware life with proprietary desktop software (office suite)	An estimate of life is taken as a basis: 4 years, so that 1/4 of users need a new PC every year	American data show approximately this life
Hardware life with open-source desktop software (office suite)	An estimate of life is taken as a basis: 6 years, so that 1/6 of users need a new PC every year	All analyses suggest that open-source desktop software makes lower hardware demands on microprocessor and RAM
Annual standard for salary costs per user (including all employer expenses and pensions)	DKK 300 000	The figure reflects the fact that many low-pay groups largely do not use IT at public workplaces, while virtually all white-collar staff with higher levels of education use IT
Annual licence fee for desktop for proprietary software (comprises client access to server, operating system and office suite)	DKK 2 000	The amount is based on an estimate that is deliberately set low
Value of user's loss of productivity in changing desktop (per year) equivalent to one working day	Every change of desktop entails a loss of productivity consisting of time spent on training instead of work and a period of relatively lower productivity corresponding to the learning curve: one working day per user per year – measured in value corresponding to 1660/1924 x 7 x annual salary ≈ DKK 1 000	Regardless which (new) desktop, time is required for user to get to know it. Converted to value as proportion of working hours in hours of work in the annual standard. Productivity with various types of software licence is assumed to be identical
Value of one day of courses measured		Calculated as a day of pay

as proportion of annual standard for pay expenditure	1% of annual standard	with corresponding addition of double for the course, altogether corresponding to 3 x 0.33% of the annual standard amount
Productivity loss of IT support staff in changing operating system software per year viewed over minimum period of four years. Standard annual amount for support indicates the additional support and maintenance costs for proprietary software, which has more frequent upgrades than open-source software.	Every change of operating system software (upgrading) is accompanied by courses for IT support with absence from the workplace and purchase of courses. Estimated additional expenditure in annual standard amount for IT support for proprietary software is DKK 100 per user	Productivity in various types of software licence is regarded as identical. Differences in bug fixes of significance for loss of productivity are not included. Additional expenditure due to change of software with training on supplier courses is set low
Hardware price for desktop computer with network card	Regardless of software, a computer price is set at DKK 8 000	This corresponds to the current offer for PCs purchased with volume discount (excl. VAT)
Additional expenditure on proprietary UNIX software server per user with 1000-user capacity efficiency per year covering support, procurement, licences and overheads for servers used for intranet, extranet and Internet tasks. Internet tasks comprise operation of internal and external firewalls, web services including caching, business-to-business web tasks and business-to-consumer web tasks	American 'total cost of ownership' difference between proprietary UNIX and open source showed USD 307, converted to DKK 2 300	On the basis of American studies, the additional expenditure is calculated per user with 1 000 users, which is converted to Danish conditions, a minimum number of servers being estimated in the public sector
Additional expenditure on proprietary UNIX software server per user with 1000-user capacity efficiency per year for servers used for more demanding 'cooperative tasks'. These comprise e-mail, common calendar, common folders and databases, threaded discussions, user-adapted applications	American 'total cost of ownership' difference between proprietary UNIX and Linux showed USD 1 150, i.e. DKK 8 625	On the basis of American studies, the additional expenditure is calculated per user with 1 000 users, which is converted to Danish conditions, a minimum number of servers being estimated in the public sector

With the above assumptions and scaling factors, we can present rough estimates of possible socio-economic gains in changing over to open-source software in specially selected general software areas in future public IT investments. The estimates can be used to show what a decision *not* to make investments in open source would entail in socio-economic losses, or the estimates can be regarded as qualified estimates of the socio-economic scope for putting open-source products to use over a shorter or longer period of time in public administration.

Table 8.5. Estimate of the socio-economic consequences of open-source options

Calculation models	**Result**	**Comment**
Annual socio-economic *additional expenditure* on proprietary desktop in the public sector compared with full change-over to open-source desktop without licence fees	DKK 900 million per year	Only the direct licence fees for a desktop workstation are included here
Annual socio-economic *additional expenditure* on proprietary desktop in the public sector compared with full change-over to open-source desktop with support licence to StarOffice software	DKK 720 million in first year and DKK 900m in subsequent years	This conditional on a StarOffice 6.0 licence for a single payment of DKK 400, which for 450 000 users is equivalent to DKK 180 million
Annual socio-economic *additional expenditure* on proprietary UNIX server	Rounded	This includes all maintenance costs calculated per user at 1 000 users for simple Internet tasks (websites, access to

compared with Linux open source server for Internet tasks	DKK 1 000 million per year	documents and printing of forms etc.) for 450 000 users
In the short term: Annual socio-economic *additional expenditure* on proprietary UNIX server compared with Linux open-source server for administrative tasks	Rounded DKK 400 million per year	This only includes 1/10 of the public users, reflecting the fact that these server tasks are not yet generally widespread (record keeping, document management etc.). The proportion will grow substantially with the spread of e-government (see below)
In the longer term: Annual socio-economic *additional expenditure* on proprietary UNIX server compared with Linux open-source server for demanding administrative tasks that completely replaces the simple server solution	Rounded DKK 3 900 million per year	Here all public users have access to demanding server tasks, reflecting a high degree of implementation of e-government
In the short term: The total socio-economic *additional expenditure* with proprietary desktop software compared with open source viewed over a 4-year period with replacement of desktop and hardware without corresponding switch for open source	Per user per year: Licences: DKK 2 000 Desktop switch: DKK 1 000 HW replacement: DKK 2 000 System switch: DKK 100 Total: DKK 5 100 per user per year. Cumulative total for the four years: DKK 9 200 million DKK 2 300 million per year	Open source products are less demanding on processor power than the proprietary Microsoft products we have seen in desktop and office suites. We assume that Windows operating system is not supplied free in this set-up
In the longer term: The annual socio-economic *additional expenditure* with proprietary desktop software compared with open source, viewed over a 12-year period with three *more* replacements of desktop and operating system and one *more* switch of hardware for proprietary platform than for open source	Per user for 12 years: HW replacement: DKK 8 000 Licences: DKK 24 000 Desktop switch: DKK 12 000 System switch: DKK 1 200 Total DKK 45 200 Average additional expenditure per user per year is ≈ DKK 3 750 Rounded on annual basis: DKK 1 700 million	Same as above but including switch of HW to open source. Same magnitude of expenditure on switch of HW is taken as basis. Any productivity effects are assumed to be identical for proprietary and open-source software over the period. We assume that Windows operating system is not supplied free with the hardware in this set-up

The total calculated economic scope for desktop, Internet servers and advanced servers in the public sector in Denmark is shown in Table 8.6 below:

Table 8.6. Socio-economic scope on changing over from proprietary software to open-source software calculated per year (DKK million).

	Short term (4 years)	**Longer term (12 years)**
Desktop	2 300	1 700
Internet servers	1 000	
e-government servers	400	3 900
Total per year	3 700	5 600

Note: The possibility of using 'thin clients' defers the replacement of hardware and will make replacement investments substantially cheaper. This will apply to both open-source and proprietary software, and the pace of replacement will be the same. This type of strategy will reduce the long-term difference between alternatives. It is estimated the difference will be reduced from DKK 1.7 billion per year to DKK 1.3bn per year.

The working group does not consider a total switch overnight from proprietary to open-source software on the desktop and on servers to be a realistic possibility. The requirements for service, support and maintenance alone rule out such a large-scale switch. On the other hand, there is good reason to look at the software alternatives in expanding (and upgrading) use of IT by the public sector to e-government.

Whether this socio-economic potential is genuinely present to the extent calculated depends partly on the degree to which open-source software has already been put to use in the public sector. As there are no figures on this, we have chosen to leave this question open. There may also be overlaps between tasks on servers, so that the socio-economic gains in simple Internet tasks and demanding administrative tasks cannot be added together, as shown here in the short term, but an attempt has been made to eliminate the significance of this by making a very low estimate of server use. As server functions will necessarily rise as e-government advances (partly due to citizen service), there is probably *underestimation* of the possible socio-economic gains with open source in the area of infrastructure in the short term.

The pace of replacement for both software and hardware has a great effect on the socio-economic framework, which is neither surprising nor incomprehensible. It is necessary to emphasise this effect, because expectations have been built up of a rate of replacement of 3-4 years for desktop software and 5-6 years for infrastructure, which, if a socio-economic yardstick is applied will be a heavy burden, unless equivalent gains in productivity can be substantiated. The latter is a contentious issue. The question is who will benefit from the doubt: the economy or the suppliers?

8.3. Conclusions

The above socio-economic gains in switching to open-source software in selected areas of IT use by the public sector are calculated with strategic considerations in mind. They therefore do not reflect a possibility of the public sector being able to bank a gain directly. Government investments in IT should on all accounts be accompanied by strategic considerations on organisational changes, which create productivity gains and ensure that these are beneficial in terms of total public expenditure.

The strategic considerations of the public authorities on e-government ought to be accompanied by considerations on what *initiatives* can advantageously be taken to acquire larger or smaller parts of the possible socio-economic gains when open source is used, because there is significant scope for this, as shown in the table above.

In this report, we have looked at the question of whether open source is a real alternative in e-government. The economic estimates show that there is great economic scope for investments in both IT staff and pilot projects in choosing open-source as an alternative to proprietary software under the applicable economic market terms in a number of non-minor areas of software.

In the existing competitive situation for infrastructure and desktop software, there is also scope for significant socio-economic gains by influencing prices and licence terms for proprietary software. This can be done by selecting open-source software in a significant number of cases and thereby establishing an alternative to the dominant industry standards. Alternatively, attempts can be made to establish open standards for a number of key tasks, but experience from standardisation work to date is that suppliers or strong concentrations of purchasers (e.g. the military, the state sector) have to come into line on a standard if it is to make a breakthrough in practice. Whatever choice is made, it will be necessary for decision-makers in the public sector to develop strategies for future IT investments with the inclusion of open-source software.

Chapter 9

Conclusions and recommendations

• Open-source software is a serious alternative
Open-source software has been put to use in many individual installations in Denmark and abroad. We have shown in a number of examples that open-source software represents a serious technical and economic alternative to proprietary software - even where there are proprietary industry standards.

Although, in reality, open-source software is based on a 'free-of-charge principle', it can be commercialised, as by far the majority of the costs of software are not accounted for by the original development but by services in connection with adaptation and maintenance. Many large private enterprises, such as Sun, Hewlett Packard, Hitachi, SAP, CA and IBM put money into developing open-source software.

In principle, the advantage with open-source software is that the user obtains a higher degree of independence from the supplier and greater freedom of choice with regard to the other software it is to be used with, because open standards are used. Finally, the available source text provides an opportunity for independent reviews of security and other aspects, and it is possible to have relatively great trust in the security of open-source products which, like the Apache web server, are based on a strong environment of enterprises and independent developers.

There are therefore strong arguments for considering open-source products in the procurement and replacement of software in the public sector.

The examples the group has studied do not have a background in a common strategy for the choice of open source in the agencies and organisations concerned. Open source has come into consideration because of individual IT managers' own assessments.

• Open source provides significant economic room for manoeuvre
The economics in a large number of open-source installations has been shown to be better than comparable proprietary software installations abroad and in Denmark, in that savings are made on licence payment and hardware procurements. In addition to this there are economic benefits in increased competition on price and quality, including security, when open-source products become widely used.

As these results do not depend on chance occurrences but on well-reasoned systematic differences in costs, they indicate that more widespread use of open-source software could have socio-economic significance.

Various estimates of the possible use of open-source software in the public sector show extensive economic effects, the annual *additional expenditure* on proprietary software today being estimated to amount to billions of Danish kroner. Opportunities for savings in the public sector on this scale should not go unnoticed.

• If e-government is to be put into effect as cheaply as possible, there is a need for a strategy
Governments in many countries have tackled the question of better service to citizens and enterprises in the light of the new opportunities for self-service via the Internet. The Danish Government has also launched e-government with a requirement for a radical improvement in the service provided by the public authorities with greatly increased use of IT as a premise.

The Government emphasises that managers in administration must develop a strategy making it possible to put the vision into effect within a very few years.

A strategy for e-government should not be based on a closed, proprietary standard in a key technology. The first reason for this is that it is unacceptable as a matter of principle for enterprises and citizens not to be able to choose between different suppliers of the software that is necessary to use the services of public authorities that are offered in the form of e-government. The second is that it is vital to the socio-economic cost-effectiveness of far-reaching e-government that a competitive situation can be established that ensures the presence of competing products. A condition that must be met for this to be achieved is that open standards are used.

Open source is based on open standards, where such standards are established and usable.

Analyses show that, in a mature software market such as that for office suites, competition does not come about of its own accord. Initiatives have to be taken for the purpose, otherwise e-government would be implemented in a monopoly situation.

The potential economic scope for open-source software will therefore not be achieved of its own accord. It requires carefully considered strategies. The working group therefore puts forward below a set of strategies aimed at e-government utilising the potential economic scope that open-source software can create in the short and longer terms:

Strategies and recommendations for selection of software for e-government
The working group recommends that the State and other authorities should jointly formulate principles and objectives for the procurement of software, on the basis of the following, among other observations:

- It is necessary for a number of decisions in relation to IT to be taken in a coordinated manner, where the State – with all ministries and agencies etc. – is capable of acting as a corporation and taking joint decisions on the basis of a multi-year planning horizon.
- Joint decisions are necessary to increase open standards, which is an essential condition to be met if a competitive situation is to be established to a greater

extent, with the use of open source as one of the options.
- In addition, central decisions are necessary to provide economic support to pilot projects and to draw up framework agreements, draft contracts etc. that can serve as offers or alternatives for local decision-makers.

In the short term

- The State must not put all its eggs in one basket. It must be ensured for all types of software that each individual administrative unit has a real choice in a competitive market.
- Open-source software must be judged on the same terms as proprietary software, and in calls for tender and other purchasing open source must be assessed on the basis of a realistic costing that takes account of all economic factors.
- Investment decisions can often be a mixture of open source and proprietary software. It is not an either-or decision, and the purchase of open source should not therefore be dictated as a general principle.
- An initial pilot project must be established in the near future in which open-source software such as StarOffice/OpenOffice is implemented in medium-sized e-government. The pilot project will be used to gather experience on the overall user-friendliness and quality of the systems, on the accomplishment of the change-over task, for example the training of users and IT staff, and on the extent and resolving of compatibility problems in connection with elecronic exchange in Microsoft formats. This experience must be put at the disposal of all other administrations.

In assessing options, special priority must be given to the value of the open source code, including the long-term value inherent in supplier independence with respect to maintenance and it being possible for security to be subjected to independent reviews.

In the longer term

- Establishment, for example within one to one and a half years, of a larger follow-up project in which a number of administrative units use open-source software, for example switching over to StarOffice/OpenOffice, and utilise previously gathered experience to reduce installation and adaptation costs.
- Preparation of a strategy for the introduction of an open standard for the exchange of word-processed documents.

The working group recommends that a standard document format is developed, firstly for problem-free exchange of documents and secondly for integration in systems used in e-government. A strategy for the introduction of an open standard for the exchange of word-processed documents is important, because there is no genuine competition at present in the desktop area, largely due to the fact that Microsoft formats also represent *de facto* standards for electronic document exchange, and among these the doc format for word processing is the most important.

Scenarios for the introduction of an open standard

In order to illustrate the strategic considerations necessary in choosing one or more document standards, we have outlined below three possible strategies in the form of three scenarios for the introduction of an open standard:

Scenario 1: A joint decision for electronic document exchange to preferentially use the XML-based format, which is used by StarOffice/OpenOffice

A decision of this kind would naturally be far-reaching and a switch-over project that would presumably take several years would be required for it to be implemented. The precise extent of the switch-over will depend on whether Microsoft develops software that can convert to and from the Staroffice/OpenOffice format. If not, the switch-over will necessitate replacing the Microsoft suite everywhere with Staroffice/OpenOffice or other formats that support the format.

Scenario 2: A joint decision to utilise two formats, Microsoft's doc format and the XML-based format of Staroffice/OpenOffice

This strategy avoids the element of compulsion in the first scenario, the aim being solely to promote the use of StarOffice/OpenOffice through rules or recommendations that equate, or in some other way make possible, but do not require, use of the format. This could, for example, mean rules in connection with the e-days that are planned by the Digital Task Force, and which are completion dates for when the public authorities are to be able to send and receive all internal communication in electronic form.[42]
On the other hand, there is a risk of persistent compatibility problems between the two formats.

Scenario 3: Development of a new, XML-based format for office software, followed by its introduction/implementation

Development work of this kind could take place under EU auspices so that more weight could be put behind the standard in the form of many users and therefore a large market for future suppliers. Denmark would presumably have a good opportunity to ensure that the standard took account of all relevant requirements, but it is not certain whether or how quickly a usable standard could be brought about.

As choosing a strategy is a demanding decision-making process, there will be a risk of a strategy for an open document standard not being formulated in reality. The consequences of this are outlined below:

Scenario X: No coordinated initiative is taken to introduce an XML-based format

The advantages of this scenario are that the risks in the others are avoided. A wait-and-see approach is adopted with regard to how the market develops,

[42] See description of project at http://www.e.gov.dk/sitemod/design/layouts/default/index.asp?pid=2130

including whether Microsoft comes up with an XML-based format by which Word documents can be exchanged and that other suppliers can handle without encountering compatibility problems. Microsoft has let it be known in meetings with members of the working group that the firm is, in principle, in favour of XML and that it is expected that MS Word will in future support XML, although this will not apply to all the features of Word.

The decisive aspect is that a strategic decision is taken on how an attempt will be made to introduce open standards on the desktop in government. This may be done with the aid of one of the strategies outlined above, or in combination.

None of the strategies should be implemented without a thorough examination of advantages and disadvantages. This naturally applies in particular to the first and most radical strategy which, on the one hand, may lead to large savings but on the other involves significant risks. The first two strategies, which entail a complete or partial commitment to the XML-based format developed in connection with Staroffice/OpenOffice, require efforts to minimise risks by first gathering experience, for instance in the form of the pilot projects mentioned above, with the StarOffice/OpenOffice format, with the whole product and with the actual switching process.

The *de facto* standard at present is identical to Scenario X. This is not a conscious choice, but rather the *absence* of an adopted strategy. The present framework agreements under *Statens og Kommunernes Indkøb* (SKI) reflect this strategy, where the decisions are *devolved* to the individual administrative units. Decisions here are often characterised by a short-sighted aversion to risk, where preference is given to living with the known drawbacks of known products over unknown drawbacks of unknown products. It is unfortunate, for example, that there has been no strong central coordination of how to respond to Microsoft's new licensing rules.

Although the new rules have far-reaching strategic consequences, in the form of both payment for new versions for which there is limited need and in the form of strengthened ties to Microsoft formats throughout the area of central government, this has not been treated as a joint, strategic decision. As a result, the State deprives itself of the advantage it would otherwise have as a large customer. A strategy to introduce open-source software would at the same time have the consequence of the public sector being in a far stronger negotiating position when the SKI agreements have to be renewed.

It is therefore not sufficient for us in Denmark to follow Britain and Germany, for example, in merely recommending that open source should be 'considered'. A more active decision must be taken in those areas where there is a *de facto* monopoly. It is necessary for decision-makers in the public sector to develop strategies for future IT investments that include open-source software.

Annexes

Annex 1

Glossary

Operating system
The most fundamental software run on a computer. All computers have an operating system, which is used among other things to start and run other programs. The operating system performs important tasks, such as receiving data from the keyboard and mouse, sending data to the screen, keeping files and folders under control and checking the various units in the machine (e.g. disks, printers etc). An operating system as a rule also contains a *user interface,* which for example enables the user to control the machine using graphic icons, windows and menus. Examples are *Windows 98, Windows 2000, Windows XP, UNIX-Solaris, AIX, MacOS, Linux.*

Server
The main computer in a network with several computers (clients). The server deals with tasks that are common to a number of clients, such as backup copying, heavy calculating tasks, control of access to the network, control of shared printers, handling of e-mail, storage of data in databases and storage of websites. The server tends to be equipped with a powerful *CPU* and has significant storage capacity. Clients can go into the server and retrieve, save or modify information. A server may be either a powerful PC or a *mainframe* and is normally linked to other servers in a larger network.

Client
Term for a computer when it is in interaction with a server. A client may be a PC.

Thin clients
When thin clients are used, all data processing and data storage take place on the server, and the client is only used for data entry and for screens.
Thin clients only use those components that are necessary to work as a client. They may be special machines without a hard disk and with limited capacity, or they may be PCs, where only parts of the PC are used as a client.

Compatibility
Term for the common use of rules, media and formats that must be complied with by the sender and recipient in a communication process, and that make it possible for these parties to carry out this process, so that the recipient can recreate the information sent.

Desktop
General term for a computer that sits on the desk, e.g. a stationary PC. The term is also used here to mean portable computers. Desktops are often used as clients in conjunction with servers.

Desktop software, infrastructure and custom built software are defined at the start of Chapter 3, page 20.

Standard
A technical specification, which is approved by a recognised standardisation body for repeated and constant use, but compliance with which is not mandatory in the legal sense, unless there is special authority for this. Standards mean that products and data formats become uniform and easy to use. Examples are the Internet protocols (TCP/IP) and the language used for websites (HTML)

De facto standard: a specification which in practice is very widespread without being approved by a recognised standardisation body. Examples are Microsoft's document formats and Adobe's .pdf formats.

Open standard: an approved standard, where the specification is publicly available.

Source code
A computer program in its original form, which has been written by the programmer in a programming language such as COBOL, C, Java, Perl, etc. The source code cannot be executed by the computer directly, and has to be translated into *machine language* by a *compiler, assembler* or *interpreter*.

Open source (or open-source software)
Term for programs that are offered so that source text is supplied with them (free of charge), where users themselves can make corrections and – depending on licence restrictions – improvements. The principle of open source is therefore not synonymous with free software. An open-source program in other words might well be transferred under licence restrictions.

Proprietary software
Programs where the purchaser is exclusively entitled to use the translated program in the form of machine language. Users cannot correct or check the source code, nor can they decide for themselves who is to continue to develop the system or fix bugs in it.

Non-proprietary software
Programs where customers have developed a system and where the customer owns the source code. The customer may leave it to other developers to continue developing the system.

TCO
Total Cost of Ownership. Statement of all the direct and indirect costs of a product (a program) throughout the life of the product from the decision to procure to decommissioning. The statement comprises both software and hardware and personnel for maintenance etc.

Annex 2

MS Office/StarOffice document exchange test

Purpose
The purpose of this test is to illustrate the degree of compatibility between StarOffice and Microsoft Office. Three types of documents have been chosen: Word, Excel and PowerPoint.

The was performed between May and August 2002.

Technical conditions
An Office suite (97-SR2, Danish edition) and StarOffice (evaluation, English edition) running on an NT platform version 4.0 were chosen to carry out the test.

The document description is based on an estimated classification of complexity. This classification is shown in Fig. 1. Where there was doubt on the classification, further reinforcement was made in the form of plus and minus (+/-).

The documents were firstly collected in one round (first test round (Annex 1 – 18) and all the documents from the working group in May-August were then tested.

Degree of difficulty/complexity (Fig. 1):

Complexity	Name	Description
Simple	A	Ordinary layout (e.g. highlighting, bold, simple tables, bullets etc.)/Ordinary calculations and formulae
Medium	B	Medium layout (header, footer, tables, pictures etc.)/long formulae, graphs etc.
Difficult	C	Heavy layout (A + B and fields (page, chapter, initials, tables of contents)/conditional formatting, list box etc.

Document description
The documents are divided into an attachment number describing which application the document comes from, a brief description of the contents, the estimated degree of difficulty and finally a description of the degree of difficulty.

Attachment	Description	Difficulty	Description
B1-word	Memorandum on digital labour-market service	B	Header, margin text, 3 pages.
B2-word	Set of tasks	B+	Header, footer, fields (page no.), graphs, tables, 9 pages
B3-excel	Calculation of training weeks	A	8 880 rows, 20 columns
B4-word	Report form, research agency	B	Complex form, 4 pages.
B5-excel	Overview of OpenSource	A	1 worksheet
B6-word	New loan, preparation for e-commerce	C	10 pages, fields, graphs, diagrams
B7-word	Compressor description	A+	Photograph, dots, water marks!!
B8-pp	PowerPoint	A	3 slides
B9-pp	PowerPoint	A	29 slides
B10-word	Description of trip	B+	Header, footer, pictures, tables, 4 pages
B11-word	Internal info	A	1 page
B12-word	Description of trip	A+	2 pages
B13-word	Minutes from Board of Technology	A+	2 pages
B14-word	Minutes from Board of Technology	A+	3 pages
B15-word	Memorandum on government IT network	B	Containers, 3 pages
B16-word	The Twinning Project	C	70+ pages
B17-word	Memorandum from Digital Task Force	B	3 pages
B18-word	Standard account for AMU	B	37 pages
B19-excel	Training-week rates	A	
B20-word	Chapter 7 – Economic analyses	A	23 pages; tables/graphics
B21-word	Attachment Microsoft licences	B	6 pages; header/tables/footnotes
B22-word	Chapter 3 Part 1	B	11 pages; header/footer/tables
B23-word	Chapter 4 educational institutions	B	2 pages; header/footer

Method
All Word attachments are printed from the MS application. The attachment is then opened in SO and saved in SO format. MS and SO are opened side by side for visual inspection. The SO (only Writer) attachment is printed and the two printouts are compared. The SO version is modified (one line is inserted on page 1) and saved in MS format. The attachment is opened in MS and reviewed.

The following test is performed:

Item/test	Description
1	Print out original (word) attachment in A4 (HP4050 Series) ?: Can the attachment be printed from Word?
2	Opened attachment in SO. (Not converted) ?: Can the attachment be opened in SO in word format?
3	Can the attachment be saved in SO format? (Saved with same attachment name + sxw ext.) ?: Can the attachment be saved as SO format?
4	Is there direct agreement between MS and SO versions? (Two windows opened with Word and SO)? ?: Is there visual agreement between Word and SO versions?
5	Print SO attachment. ?: Can the attachment be printed?
6	Items 1 and 4 compared ?: Is there visual agreement between the two printed pages?
7	Edited in SO version on page 1 (one line inserted), saved as original attachment (format) ?: Can it be saved in original format?
8	Attachment opened in Word/Excel/PP, is item 6 included? ?: Has the editing gone, though?
9	Comparison of attachments after Item 7 with Item 1. ?: Has the editing destroyed other formatting/content etc.?

Degree of conversion:

Complexity	Name	Description
No loss	1	Attachments are identical (if differences have arisen due to fonts, printer settings etc., these are disregarded).
Layout loss	2	Attachment is changed in relation to original layout, e.g. pictures are displaced, margin is moved or similar. The degree of conversion is described for each individual attachment.
Information loss	3	Information has been lost (text, numbers, graphs etc. have disappeared) The degree is described for each individual attachment.

Attachment	X	1	2	3	4	5	6	7	8	9
B1-word	B	Yes (1)	1	1	1	1	1	1	1	1
B2-word	B+	Yes (1)	1	1	2(3)	1	2(3)	1	1	1
B3-excel	A	NA	1	1	1	NA	NA	1	1	1
B4-word	B	Yes (1)	1	1	1	1	1	1	1	1
B5-excel	A	Yes (1)	1	1	1	1	1	1	1	1
B6-word	C	Yes (1)	1	2	2	1	2	1	1	1
B7-word	A+	Yes (1)	1	1	1	1	1	1	1	1
B8-pp	A	NA	1	1	1	NA	NA	NA	NA	NA
B9-pp	A	NA	1	1	1	NA	NA	1	1	1
B10-word	B+	Yes (1)	1	1	1	1	1	1	1	1
B11-word	A	Yes (1)	1	1	1	1	1	1	1	1
B12-word	A+	Yes (1)	1	1	1	1	1	1	1	1
B13-word	A+	Yes (1)	1	1	1	1	1	1	1	1
B14-word	A+	Yes (1)	1	1	1	1	1	1	1	1
B15-word	B	Yes (1)	1	1	1	1	3	1	1	1
B16-word	C	No (3)	1	1	1	1	NA	1	1	NA
B17-word	B	Yes (1)	1	1	1	1	1	1	1	3
B18-word	B	Yes (1)	1	1	3	1	3	1	1	1
B19-excel	A	NA	1	1	1	NA	NA	1	1	1
B20-word	A	Yes (1)	2	2	2	2	2	2	1	1
B21-word	B	Yes (1)	1	1	1	1	1	1	1	1
B22-word	B	Yes (1)	1	1	1	1	1	1	1	2
B23-word	B	Yes (1)	1	1	1	1	1	1	1	1

Comments

NA = Not applicable in the tested document.

Attachment 1.
No comments

Attachment 2-word.
Item 4, OLE *) objects faulty
Item 6, OLE *) objects faulty

Attachment 6-word.
Item 4, Logo on page 1 missing.
Item 4, OLE *) objects faulty
Item 6, OLE *) objects faulty

Attachment 15-word.
Item 4, missing heading on table, page 2.
Item 7, same heading missing.

Attachment 16-word.
The document cannot be printed by Word (An application error has occurred (WINWORD.EXE)). Attempts also made to print from Windows95, Windows98, without success. The document was returned to the user, who was unable to print the document either.

Attachment 17-word.
Item 4, in bulleting: first word in sentence missing, bullets 2 and 3 merged.

Attachment 18-word.
Item 4, on a diagram axes/background drawn but curves missing.
Item 5, diagram lacking curves.

Attachment 20-word.
Items 4 & 7, OLE *), on last page text is positioned on top of text on pages 5, 7, 9, 10 and 22. On examination of formatting in MS Word: the anchor is not locked and 'Apply over text' and 'Move object together with text' checkboxes are selected.

Attachment 22-word.
Item 4, footers are present, but there is no page total in them.
Item 7, number of pages missing.

Number of documents contained:

Word: 18/17
Excel: 3
PowerPoint: 2

If the number is not always identical for each text in all cases, this may be due to the test element not being relevant and therefore not being carried out. See comments on each document.

Conclusions
On the basis of the degrees of conversion and remarks, it is judged that information is generally not lost. On the other hand, layout will more often be lost, particularly positioning of graphics (anchors). Loss of information has only arisen in connection with OLE objects.

*) OLE = (Object Linking and Embedding) is when another program is embedded in the program being used. An independent spreadsheet may, for example, be embedded in a text document.

Annex 3

Overview of Microsoft licences

The working group has used Microsoft software as a comparison alternative at several places in the report because Windows and MS Office on clients are a *de facto* standard and because exchanging of files usually takes place using Microsoft formats.

We therefore consider it necessary to present a brief description of Microsoft licences.

Microsoft has radically altered its licence policy and installation requirements in recent years. This applies in particular to the following items:

- upgrade versions of individual products no longer exist. It is only possible to purchase versions at the agreement price
- the right to upgrade to the latest version can be obtained for an annual licence at 25-29% of the agreement price.
- all institutional agreements are for three years
- Microsoft is committed to agreements that cover the customer's entire need and where the customer pays an annual fixed sum, regardless of the extent to which what is possible under the agreement is utilised

Public institutions in Denmark to a great extent purchase on the basis of the framework contact for 2001-2004 entered into between the local-authority purchasing service *Statens og Kommunernes Indkøbs Service* (SKI), Microsoft and a number of suppliers of Microsoft software. The SKI agreement has been put out to tender, and only the possible variants laid down in the agreement exist, while the agreement only covers the suppliers stated in the annex to the agreement. The framework contract ceases on 31 March 2004, and it is not possible to make agreements with a longer term under the framework agreement. Public institutions that want a three-year agreement outside the period of the framework contract therefore have to enter into a special contract on the same terms as private enterprises. Microsoft generally offers a discount for volume, and as the Danish government sector is a customer making large purchases under the SKI agreement, with partial joint responsibility, equivalent discounts are obviously obtained.

The SKI agreement covers specific suppliers, who themselves set a discount on the basis of the SKI agreement. The discounts vary from supplier to supplier and should be viewed in conjunction with the support and service provided.

It is one of the terms of the SKI agreement that it is only available to members of SKI. Prices etc. are not publicly available. We have therefore taken bids from suppliers of software as our basis in the report.

The factual description of the licences and the software requirements has been submitted to Microsoft Denmark, and they have not expressed any objections to it. The judgements and calculations of consequences in the report are obviously the working group's own responsibility.

Microsoft judges that outside the educational sector 90-95% of all public procurement of Microsoft software takes place under the SKI agreement. A number of small institutions purchase the Microsoft Open Licence. Among public institutions, around one-fifth lease and four-fifths purchase. The proportion of leasing is slightly higher in the private sector.

Volume licence options

Microsoft agreements are generally entered into for a period of three years. There are several levels of volume discount for each of the agreements. All Microsoft agreements assume all the customer's PCs having a professional 32-bit Windows operating system, which means that the agreements only contain Windows upgrading.

In addition to the general agreements there are two forms of agreements which may be purchased separately where appropriate:

Product Support Service

- it is possible to purchase a support agreement for one's products. The support agreements can be signed with several different levels of cover and over one, two or three years

Software assurance (SA)

- Software Assurance (SA) is a licence for upgrading the licences held, whether they are owned or leased
- upgrading applies to the latest version of the product at any time
- SA can only be purchased if one has the latest version of a Microsoft product at the time of the contract
- SA can be purchased as a separate three-year licence and will typically cost an annual fee of 25 to 29% of the purchase price
- if an SA agreement is renewed after three years, the rate is slightly lower

The price for the three forms of licence below (OSL, EA and ESL) is based on the number of 'qualified PCs' and not on the number of products actually installed. These licences therefore provide an incentive for a high degree of standardisation of software on clients. A 'qualified PC' is any sufficiently large client PC, regardless of whether it runs Windows or MS Office or not.

For EA, OSL and ESL, Software Assurance is included in the term of the contract.

Open Subscription License (OSL)
- entails a right to use the software on the platform, i.e. the licences are not owned when the agreement comes to an end
- if one wishes to take over right of ownership when the agreement comes to an end, 1.5 times the annual price per OSL product has to be paid

Enterprise Agreement (EA)
- purchase of software on the PC/Mac, i.e. ownership of licences is retained after the contract expires

Enterprise Subscription Licence (ESL)
- covers right to use software on the platform, i.e. one does not own the licences when the agreement comes to an end
- if one wishes to take over right of ownership when the agreement comes to an end, 1.5 times the annual price per ESL product has to be paid

The Select agreement differs from the above in that it is concerned with purchases of individual licences on the basis of a price list:
- the prices of products that can be purchased under the Select agreement are subject to change
- the Select agreement can be combined with a Software Assurance agreement, which ensures that upgrading is possible if one has the latest version of the Microsoft product. The annual licence for an agreement of this kind varies from product to product and is between 25% and 29% of the price of the product
- the Select agreement can be combined with Support Service as a separate contract
- the price under Select is based on the annual number of purchased 'points', where various programs are included with varying points

Forms of licence for educational institutions
Microsoft offers various special, very cheap, licences to educational institutions. They are broadly equivalent to the terms of an ESL. The agreements are adapted to the conditions that apply to the various types of educational institutions. The Campus agreement, for example, only applies to the universities. The Select agreements for educational institutions are generally priced much lower than the general Select agreement. In turn, there are different types of agreements for different educational institutions. Academic Select, for example, is only applicable to universities.

These agreements also include special definitions of how many licences are paid for in relation to the number of users.

The educational institution purchases through a dealer, and here too there are discounts of varying extent depending on the supplier, matched to the level of support and service.

Which form of licence is cheapest for a public institution?
This question depends on the strategy adopted by the institution in relation to software. This selection includes a number of major factors, including:
- the need for support, support being included in ESL, EA and OSL
- the need to be able to upgrade the software, Software Assurance being included in ESL, EA and OSL. This need may be a major one if MS Office is used as a component part of a larger system. Prices under the agreement with SA are higher than a three-year agreement
- the chosen rate of replacement of Windows and Office
- the desire to have right of ownership of the software in the event of cessation of an agreement, leasing being 15-18% cheaper than purchasing for the same volume

The additional cost has to be viewed in conjunction with the needs of the specific institution for support and the right to upgrade. If upgrading every other year is chosen, for example, the annual costs are roughly equal for all the forms of licence. Under the Select agreement, it means that licences to new software have to be purchased every other year.

The amount of the difference for the educational sector is not great, but as a result of the low price the percentage difference is substantially greater.

Annex 4

Age of PCs that can be used in latest version of Office suites

Microsoft Windows + Office suite
Microsoft has generally come along with new versions of Windows every two to three years since 1995, and MS Office every two years since 1995. Every time Microsoft has introduced new versions, the system requirement has been greater than for the previous version. From the introduction of Windows 95 up to the present, it can be said that PCs must not be more than three to four years old if they are to run reasonably well at the time of the introduction of the latest version of Windows and Office.

If Microsoft continues as it has in the last seven years, a strategy of using Microsoft's latest products as quickly as possible will mean having to replace one's PCs every three to four years. This assumes, however, that all software is run on the client.

Studies from the United States and Europe show that the majority of customers upgrade at a substantially slower rate, with only around one in eight customers following Microsoft. Among the slower ones, around one in four customers upgrade after five to six years.

System requirements for Microsoft plus Office
Microsoft indicates system requirements. General experience is that the programs can run on a PC with minimum requirements, but that practical use requires more. Experience shows that the machines as a rule have ample hard-disk space, but that the installed RAM is on the low side. Good performance necessitates installed RAM over and above system requirements as stated by Microsoft.

PC Magazine has assessed the system requirements of Microsoft software at what the magazine calls 'decent performance'. We have therefore taken as our basis PC Magazine's assessment of requirements for RAM.[43]

[43] http://www.pcmag.com/article2/0,4149,2017,00.asp (12/08/2002)

Microsoft's minimum requirements in the combination of the latest version of Windows and the latest MS Office suite.								
	1995	1996	1997	1998	1999	2000	2000	2001
Windows	95	95	95	98	982E	98ME	2000	XP
Office	95	95	97	97	2000	2000	2000	XP
System requirements								
Processor	486	486	486	486	P I	P I	PI	PIII
MHz	25	25	40	66	66	150	133	300
RAM (MS)*	8	8	16	24	32	48	64	128
RAM (PCM)**				64	64	64	128	256
HD	100	100	150	300	450	750	1.3	2GB
PC newer than	Mid 1993	Mid 1993	Early 1994	Late 1994	Late 1995	Late 1996	Early 1997	Early 1998
Age of PC	< 2 years	< 3 years	< 3 years	< 4 years	< 3½ years	< 3½ years	< 3½ years	< 3 years

* Microsoft minimum requirements have been used on the RAM (MS) row.
** PC Magazine's requirements for decent performance have been used on the RAM (PCM) row.

Age of PC (PC newer than) shows the maximum age a typical PC (price category DKK 10 000) may be when the latest versions of Microsoft software are introduced. We have taken as our basis the requirements in relation to performance drawn up by PC Magazine. More RAM can be installed in old machines to prolong their useful life, but not so that this exceeds around four years.

StarOffice 6.0 – OpenOffice 1.0
Regardless of whether Linux or Windows (95, 98, NT, 2000, XP) is used as the operating system, the requirements are modest.[44]

- Pentium-compatible PC or later
- 64 MB RAM
- 250 MB hard disk space
- VGA monitor or higher (256 colours, 800 x 600)

Both can therefore run on a machine purchased after the second half of 1995.

[44]http://wwws.sun.com/software/star/staroffice/6.0/index.html
http://www.openoffice.org/dev_docs/source/sys_reqs.html

Annex 5

Use and development of open-source software

In June 2001, PLS Rambøll conducted a study on the use and development of open-source software (OSS) in public institutions and private enterprises in Denmark. The study was carried out on behalf of the Danish Patent and Trademark Office and the Danish Board of Technology, and this section will present an account of the results of the study.[45]

Purpose
The aim of the study was to

- survey the use of OSS in Danish business and in the public sector
- survey the participation of Danish business and the public sector in the development of OSS

The study was undertaken as an Internet-based questionnaire study among private enterprises and public institutions, and the respondents are the people responsible for IT in the enterprise or institution.

Awareness of open-source software
Around half (53%) of all Danish enterprises and public organisations have never heard of OSS, and among those that are aware of OSS only 8% use it. Awareness of OSS therefore does not result in increased use of OSS.

The study also shows that awareness and use of OSS are more widespread among public institutions than in private enterprises. 61% of public institutions are aware of OSS, and 16% use OSS, against 38% with awareness of and 8% using OSS in private enterprises.

In addition, it may be found that awareness of OSS is greater, the larger the enterprise or organisation. This is probably due to larger enterprises often having a separate IT department with expertise in the area.

Some of the awareness of OSS is due to enterprises and organisations having heard about specific Open Source programs, for example Linux, which is the most widely known and used application in Denmark, but without their knowing about the principles underlying OSS.

Considerations on the use of open-source software
Those enterprises and public organisations that have heard about OSS without using it (39%) were asked about their thoughts regarding the future use of OSS.

Use of OSS looks likely to spread over the next decade. Thirteen percent of enterprises and organisations that have heard of, but do not use, OSS anticipate that they will be using it within the next two years. Another 4% anticipate that they will be using OSS in the longer term, and although these figures do not take account of the over-representation of IT enterprises in the study, it does reflect the fact that there are specific plans to introduce OSS in many places.

The study additionally shows that the two areas where most enterprises and organisations expect to use OSS in the future are

- server software (71%)
- operating systems (65%)

OSS at present is most widely used as server software and operating systems, and it is therefore in these two areas that expectations of increased use in enterprises and organisations in the future are greatest. By comparison, only around one-third expected to use OSS as office programs in the future.

Those enterprises and organisations that did not expect to use OSS in the future (76%), gave their most important reasons as

- lack of time and resources (41%)
- lack of compatibility with the programs used by cooperating partners (37%)
- lack of awareness of OSS (30%)

A switch to using OSS is characterised at present by there being fewer economic costs in procurement and enhancement, but more awareness of the software on the part of IT staff being required. This may explain why it is time and resources that are the greatest barrier to using OSS and why lack of awareness of OSS ranks third.

Among reasons why enterprises and organisations are considering using OSS, economic reasons in the form of lower procurement costs weigh most heavily (62%), followed by supplier independence (37%). Other factors that weigh heavily are freedom to enhance, higher reliability, lower operating costs and better opportunities to fix and modify the software. It is therefore freedom and economics rather than security and ideology that are the factors that count when use of OSS is considered.

Those enterprises and organisations that do not use OSS have been asked to assess what factors could induce them to use OSS. The factors most commonly mentioned are

- widespread use of one standard for file exchange (34%)
- development of programs (29%)
- more user-friendly end-user programs (27%)

In addition, around 25% state that building up skills among IT staff, greater awareness of OSS in the enterprise or the organisation and better external support are of great significance for the use of OSS in the future.

Use of open-source software
The enterprises and organisations that use OSS (8% of all respondents in the study) mainly use the software in three areas, namely

[45] The report can be downloaded from www.tekno.dk.

- IT
- development
- production and operation

While private enterprises and public organisations follow each other closely in the area of IT and the area of production and operation, there is a wide difference in the use of OSS in the area of development, where 62% of private enterprises use OSS for development, compared with only 25% of public organisations.

A relatively small proportion of the total number of respondents in the study use OSS. On the other hand, more than half (64%) of those who already use OSS anticipate increased use within the next two years. It is mainly in applications where OSS is already used that enterprises and organisations expect to increase their use. OSS is primarily used at present as server software and operating systems.

In the study, those enterprises that already use OSS identify lower procurement costs and causes that are related to quality advantages with OSS as the main reasons why they have started using OSS.

The primary benefits enterprises and operations have attained by using OSS were likewise related to economics and quality benefits:

- lower procurement costs (54%)
- better opportunities to fix bugs and modify the software themselves (35%)
- higher reliability and operational stability (35%)
- lower operating costs (34%)
- freedom to enhance the software (34%)

The number of enterprises that have experienced benefits from using OSS generally exceeds the number of those that have experienced drawbacks. The primary drawbacks are

- lack of compatibility (26%)
- lower user-friendliness for end-users (21%)
- inability to obtain the programs needed (16%)

The study additionally shows that procurement and identification of OSS entails net savings, while external support and training of IT staff entails additional costs. The impact on IT resources in enterprises and organisations is more or less cost-neutral.

Development of open-source software

As well as developing OSS, a number of enterprises have started developing products and services in which OSS are included. Of the 8% of the respondents who use OSS, only 13% do not use OSS for development.

The study thus suggests that OSS is still something that is used by a fairly small circle of enterprises and organisations that have a special understanding of the area and meet particular conditions for the use of OSS.

Main features of the study

The study shows in the main that although knowledge of OSS is reasonable, only a small proportion of private enterprises and public organisations in Denmark use OSS, and that its use is more widespread among the public organisations.

The study also shows that it can be anticipated that OSS will be more widely used over the next few years, as a smaller proportion of those enterprises that do not use OSS expect to do so in the next two years, while a larger proportion of the enterprises that already use OSS expect to increase their use of it.

The decisive reasons why OSS is used or is not used mainly relate to economic savings, awareness of OSS, compatibility, the development of programs and user-friendliness.

Annex 6

Available reports on open-source software

The actual application of open-source software is poorly documented, both in Denmark and internationally. Some consultancy and public reports that shed light on the problems from a theoretical point of view are in circulation. In its investigative and analytical work, the working group has taken account of the following reports on open-source software.

Danish reports:
Open Source i Danmark – udvikling og anvendelse. Report drawn up by E-Source Development ApS for the Danish Patent and Trademark Office, 2001.

Anvendelse og udvikling af Open Source Softvare. Report drawn up by PLS-Rambøll for the Danish Patent and Trademark Office and the Danish Board of Technology, October 2001. Report discussed in Annex 5.

Open-source software i offentlige institutioner. Report from conference held at the IT University of Copenhagen on 30 April 2001.

Casestory tilsendt Statens IT-råd. Om Forbrugerinformations open source strategi. Danish Consumer Information Centre, April 2001.

Open-source software i Nordjyllands Amt. EDB- og Informatikkontoret, December 2000.

International reports:
Total Cost of Ownership for Linux in the Enterprise. Robert Francis Group, July 2002.

Opening the Open Source Debate. A White Paper. Alexis de Tocqueville Institution, June 2002.

Pooling Open-source software. IDA, European Commission, DG Enterprise, June 2002.

How to Avoid Pitfalls and Save Money With Linux Servers. Research Note, Gartner Research, June 2002.

Free/libre and Open-source software: Survey and Study. FLOSS, International Institute of Infonomics, University of Maastricht, June 2002.

The Changing Office Productivity Application Market. Research Note, Gartner Research, March 2002.

Open Software & Open Standards in South Africa. A Critical Issue for Addressing the Digital Divide. National Advisory Council on Innovation, Open Software Working Group, January 2002.

ICT Infrastructure for Primary Education. Siceroo, January 2002.

Slutrapport om huruvida programpaketet OpenOffice.org och operativsystemet Linux lämpar sig som standard för arbetsstationer inom Åbo stad. IT Department of the City of Turku, December 2001.

Free Software/Open Source: Towards Maturity. Upgrade, The European Online Magazine for the IT Professional, Vol. II, No. 6, December 2001.

Open-source software. Use Wthin UK Government. Draft for Public Consultation, Cabinet Office, December 2001.

The Role of Linux in Reducing the Cost of Enterprise Computing. An IDC White Paper, November 2001.

Analysis of the Impact of Open Source Software. QinetiQ, October 2001.

Study into the use of Open-Source Software in the Public Sector: Part 1. The OSS Fact sheet. Part 2. Use of Open Source in Europe. Part 3. The Open Source Market Structure. IDA, European Commission, DG Enterprise, June 2001.

Free Software/Open Source: Information Society Opportunities for Europe. Working group on Libre Software (appointed by the Information Society Directorate General of the European Commission), Version 1.2, April 2000.

Åpen programvare. Anvendelighetene av Linux og åpen programvare i statslig forvaltning. Statskonsult, March 2001.