David vs. Goliath: Responses to Domination Strategies in PC and Server OS Markets

Vedat G. Diker and Jochen Scholl
Nelson A. Rockefeller College, University at Albany - SUNY
Milne 300, 135 Western Avenue, Albany, NY 12222, USA
Tel: (518) 442 3937 Fax: (518) 442 3398
vd7606@csc.albany.edu JocScholl@aol.com

Abstract

Can free, open source, or non-proprietary software gain influence and market share in a monopoly situation as it exists in today's PC and server markets? The paper tries to capture major dynamics in these markets and presents the surprising finding that Microsoft's dominance may be gone in the not too distant future.

I. Problem Identification

In the modern PC and server software arena, the position of the market dominator Microsoft seems to be perpetual. Microsoft controls more than 90 percent of the markets of Operating Systems and of major productivity tools. However, as examples such as Acrobat Reader / Postscript and Java/Sun suggest, there are opportunities to establish areas that are not or not completely under the control of the market leader. Recently, the free or open source software movement known under the names of GNU (which stands for GNU Not UNIX) and Linux has come to public and even Wall Street Journal first page attention.

The philosophy underlying the free software movement is radically different from proprietary approaches. In using a combination of copyright, patent, and trade secret legislation proprietary software vendors restrict licensees of their products to very specific and even time limited uses. Free software in contrast is free, i.e., its source code is open, is can be widely shared and copied infinitely without restriction. It can be modified and redistributed by anyone. Many scientists and researchers feel that proprietary software stifles free academic research and the progress of science as such, and hence favor free software approaches. Free software does not mean free of charge, however. Rather than charging premiums for the right to use a piece of software as in the case of proprietary software, free software charges are possible only for add-on services such as installation, packaging, maintenance, troubleshooting etc. In other words, not only the philosophies of free and proprietary software but also the business models are radically different. There are more than 10 million users of free software as opposed to some 250 million users of proprietary software as of this writing. Free software already has the highest share in the market of web servers. Can the free software movement and its business model represented by companies such as RedHat, Caldera, Slackware, SuSe and others become even more successful? What can Microsoft do about this? Will proprietary software maintain the upper hand in this battle?

II. Model Structure

II. A. Overview

The OS (Operating System) Market Model is an effort toward modeling and simulating the dynamics of the current PC/Server Operating Systems markets. The model simulates the consequences of the competition between the professionally marketed, “corporate built”, proprietary Microsoft OSs (Windows 95/98/NT etc., labeled “Windows” or “MS Windows” below) and a freely distributed, “alternative”, open source OS (GNU/Linux) that is being built by numerous “professionally unrelated” developers spread over the world. The model consists of four major sectors, three of which represent the broad behaviors of three main stakeholders in the market, namely OS Developers, Application Developers and OS Customers (PC Customers). Accordingly, these sectors are called OS Developers Sector, Application Developers Sector and OS Customers Sector. The fourth sector is the Market Effects Sector through which the effects of market share between two OSs and domination efforts by Microsoft.

The OS Developers Sector reflects the dynamics that determine the number of developers working on developing the two OSs and the system capabilities of these OSs.

Application Developers Sector is the sector where the dynamics that govern the number of developers that work on developing applications for the two OS platforms and the pool of applications that run on these OSs.

In OS Customers Sector, the number of PCs in use, the portions of this PC pool on which the two systems are installed are calculated, and the choosing behavior of PC users and the support of PC vendors are reflected.

A diagram of sectors and intermediary variables is given in Figure 1.

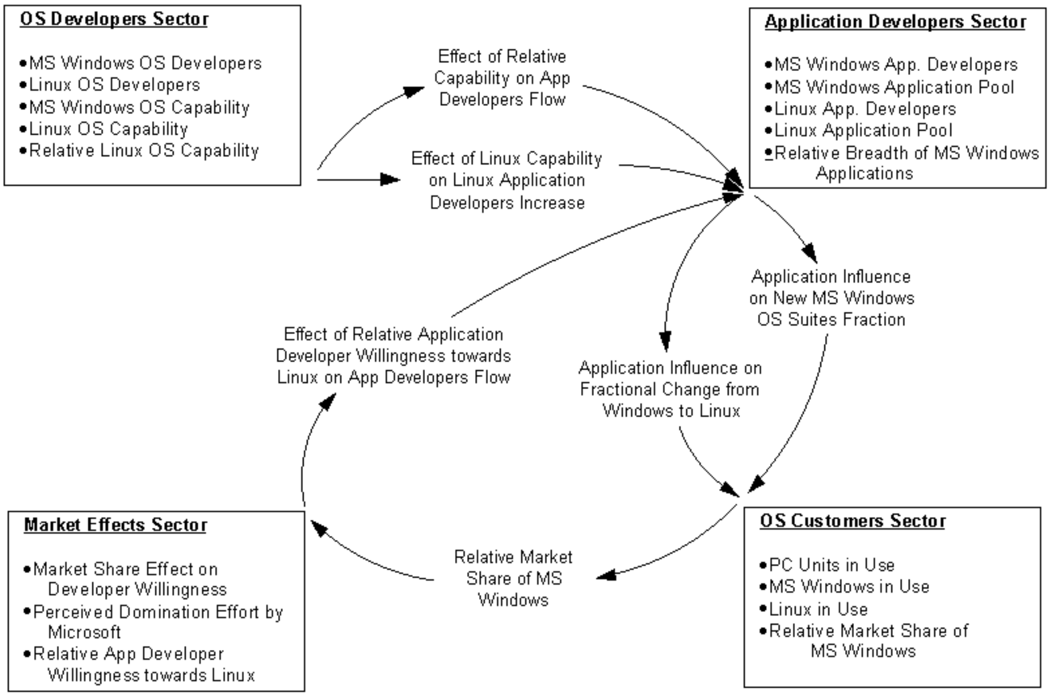

OS Developers Sector
•MS Windows OS Developers
•Linux OS Developers
•MS Windows OS Capability
•Linux OS Capability
•Relative Linux OS Capability
Effect of Relative Capability on App Developers Flow
Effect of Linux Capability on Linux Application Developers Increase
Application Developers Sector
•MS Windows App. Developers
•MS Windows Application Pool
•Linux App. Developers
•Linux Application Pool
•Relative Breadth of MS Windows Applications
Application Influence on New MS Windows OS Suites Fraction
Effect of Relative Application Developer Willingness towards Linux on App Developers Flow
Application Influence on Fractional Change from Windows to Linux
Market Effects Sector
•Market Share Effect on Developer Willingness
•Perceived Domination Effort by Microsoft
•Relative App Developer Willingness towards Linux
Relative Market Share of MS Windows
OS Customers Sector
•PC Units in Use
•MS Windows in Use
•Linux in Use
•Relative Market Share of MS Windows

Figure 1. Sectors and Intermediary Variables.

II. B. Sectors

II. B. 1. OS Developers Sector

The main stocks in this sector are ‘Microsoft OS Developer Employees’, ‘GNU/Linux OS Developers’, ‘Windows OS Capability’, and ‘GNU/Linux OS Capability’.

‘Microsoft OS Developer Employees’ represents the number of Microsoft employees that work on developing Microsoft’s OS platforms. This number increases 1.1% each quarter, and being multiplied by ‘MS Windows OS Developers Productivity’, gives the ‘Net Increase in MS Windows OS Capability’. ‘Net Increase in MS Windows OS Capability’ adds to ‘MS Windows OS Capability’, which represents the system capability of Windows. The unit of system capability is defined as OSCU (Operating System Capability Unit), which is an abstract construct. It represents the power of an operating system platform embodied by its components, its architectural capabilities, its HW scope, its programmability, its usability, and its elegance (in an abstract manner). The more capable an OS becomes, the more and the more diverse applications can be programmed and run on it. ‘MS Windows OS Developers Productivity’ is a table function of ‘Microsoft OS Developer Employees’. It is assumed that productivity of each employee decreases as the number of employees increases.

‘GNU/Linux OS Capability’ increases through ‘Net Increase in GNU/Linux OS Capability’, which is determined by the number of respective developers and their productivity, as in the case of ‘MS Windows OS Capability’. Here again, the productivity of each developer decreases as the number of employees increases. There is a difference in calculating ‘GNU/Linux OS Developers Productivity’, however. The table function is multiplied by the total effect of scenario elements that are explained in the following paragraphs.

The number of ‘GNU/Linux OS Developers’ increases through the flow ‘Net Increase in GNU/Linux OS Developers’. This flow is a function of maximum possible increase and the effect of a group of scenario elements. Maximum possible increase is assumed to be the net change in the number of GNU/Linux OS developers if all scenario elements are at their highest possible level, thus representing the ‘best possible case’.

The scenario elements that influence ‘Net Increase in GNU/Linux OS Developers’ are ‘Communication’ such as the Internet as low price backbone for idea exchange and program (re-)distribution, ‘Technical Features’ such as stability, adaptability, scalability, and elegance, and ‘Philosophy’ such as open source/ free software and priceworthiness.

There are two main feedback loops in this sector. The first loop is the reinforcing (positive) loop that involves the variables ‘GNU/Linux OS Capability’ and ‘GNU/Linux OS Developers’. As the number of developers that work on developing GNU/Linux increases, so does ‘GNU/Linux OS Capability’, and as ‘GNU/Linux OS Capability’ increases more and more OS Developers choose to work on GNU/Linux, thus increasing ‘GNU/Linux OS Developers’. (Figure 2).

The second main loop in this sector is the one that involves ‘GNU/Linux OS Developers Productivity’, as well as ‘GNU/Linux OS Capability’ and ‘GNU/Linux OS Developers’. It is assumed that an increase in number of ‘GNU/Linux OS Developers’ has a decreasing effect on the productivity of each individual developer, so this loop is a balancing (negative) one. (Figure 2).

The main outputs of this sector to other sectors are 'GNU/Linux OS Capability' and 'Relative GNU/Linux OS Capability', which is the ratio of GNU/Linux OS capability to the total of the capabilities of the two OSs. 'GNU/Linux OS Capability' influences 'Effect of GNU/Linux Capability on GNU/Linux Application Developers Increase', which goes into the Application Developers Sector and influences 'Net Increase in New GNU/Linux Application Developers' there. 'Relative GNU/Linux OS Capability' influences 'Effect of Relative Capability on Application Developers Flow'. That variable also goes into Application Developers Sector, and influences 'Fractional Application Developers Flow from MS Windows to GNU/Linux', the fraction of application developers that decide to move from building Windows applications to building GNU/Linux applications, or vice versa.

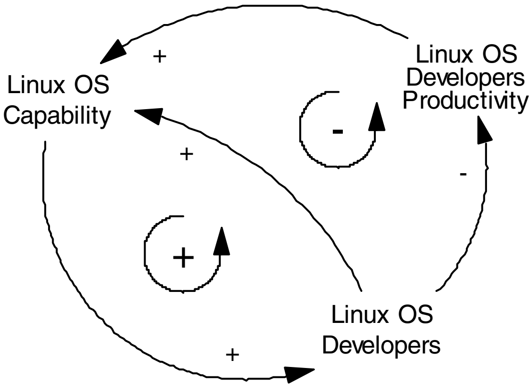


Figure 2. Main loops in OS Developers Sector.

II. B. 2. Application Developers Sector

There are four main stocks in this sector: 'MS Windows Application Developers' and 'GNU/Linux Application Developers' represent the number of developers who work on building application for MS Windows and GNU/Linux respectively. 'MS Windows Application Pool' and 'GNU/Linux Application Pool' represent the amount of finalized/released applications that run on each of these two OSs.

The unit used to represent the amount of such applications that run on a given OS is named VGAU (Value Generating Applications Unit). The 'quantity' that this unit represents can be regarded as a 'convolution' of the number of applications run on a given OS, the 'significance' of these applications in terms of the potential value that can be generated by using them, and the 'support potential' of developer companies that stand behind these applications. So the fancy program "XYZ" written by moonlight hacker Johnny Foobar is not seen as Value Generating Application under this definition, while MS Excel or SAP R/3 surely is.

'Pools' of applications that run on each OS increase through new application developments, which is a function of the number of developers and their respective productivity. The productivity of GNU/Linux Application Developers is assumed to be higher (roughly twofold) than those of MS Windows Application Developers, due to factors such as free and open source programming and the rapid and numerous feedback on public program code.

The application pools decreases through application obsolescence. MS Windows applications are assumed to get obsolete sooner, due to market conditions and frequently updated OS and application versions that are incompatible with previous versions in a number of cases. MS Windows applications have an average life span of 12 quarters, while the average life span of GNU/Linux applications is 20 quarters. This also reflects the fact that "revving up" open source code applications runs more smoothly than proprietary (hidden) source code applications.

It is assumed that the total number of application developers increases by 1.5% each quarter. A certain fraction of the net increase in application developers, which is determined by the effect of GNU/Linux OS Capability and the combined effect of a number of scenario elements, goes into 'GNU/Linux Application Developers' stock. The rest of the net increase goes into 'MS Windows Application Developers'.

The scenario elements that influence 'New GNU/Linux App Developers' are 'Communication', 'Technical Features', and 'Philosophy'. The Communication effects here are the same as on the 'Net Increase in GNU/Linux OS Developers' in the OS Developers Sector. However 'Technical Features', and 'Philosophy' elements have slightly different meanings (Technical Features) and different dimensions in this sector than in the OS Developers Sector, and are thus represented by different variables. Whereas Technical Features refer to elegance, stability, adaptability, and the scalability in the OS Developers Sector, they refer to robustness, maintainability, compatibility, and priceworthiness in the Application Developers Sector.

Another structure that influences the distribution of application developers between the two stocks is the flow of developers from MS Windows to GNU/Linux, and vice versa. 'Net Flow from MS Windows to GNU/Linux Application Developers' is determined by 'Fractional Application Developers Flow from MS Windows to GNU/Linux', which is a function of relative breadth of applications run on each OS, relative capability of the two OSs, market driven willingness of application developers, and the effects of the scenario elements explained above.

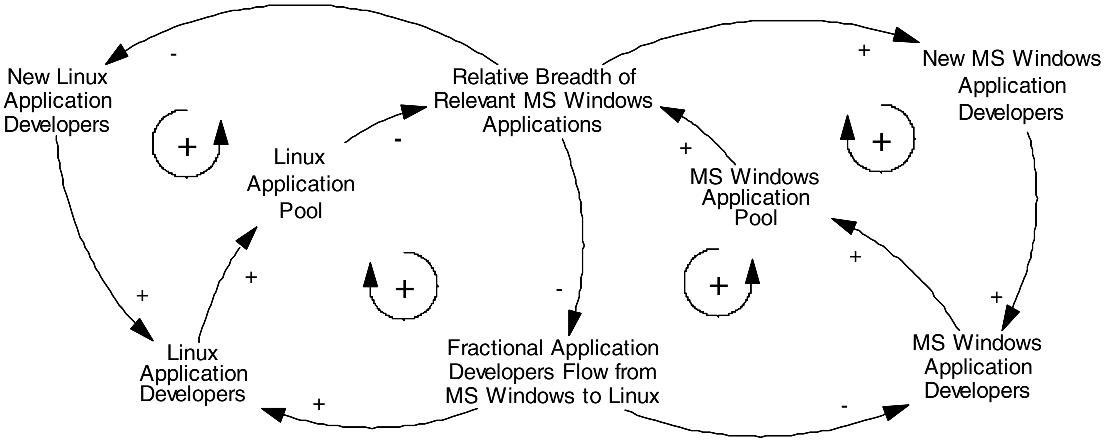

New Linux Application Developers
Relative Breadth of Relevant MS Windows Applications
New MS Windows Application Developers
Linux Application Pool
MS Windows Application Pool
Linux Application Developers
Fractional Application Developers Flow from MS Windows to Linux
MS Windows Application Developers
+
-
+
-
+
+
+
+
-
+
-
+
+
+
+
+

Figure 3. Main loops governing Application Developers Sector.

This sector is governed by four positive feedback loops. Two positive loops reinforce the number of GNU/Linux application developers and GNU/Linux application pool through 'New GNU/Linux Application Developers' and fractional flow from MS Windows application developers to GNU/Linux application developers. Another two loops reinforce the number of MS Windows application developers and MS Windows application pool through 'New MS Windows Application Developers' and fractional flow from GNU/Linux application developers to MS Windows application developers. (Figure 3).

II. B. 3. OS Customers Sector

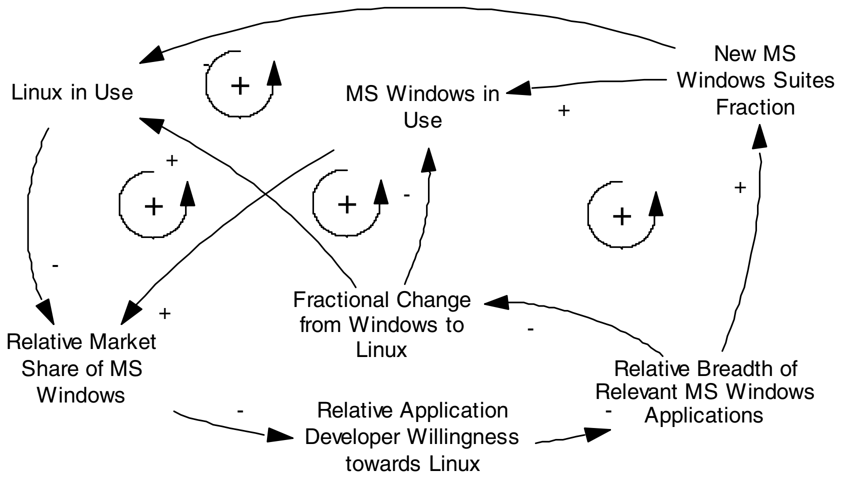
This sector represents the main OS market dynamics. The two main stocks are 'MS Windows in Use' and 'GNU/Linux in Use'. These stocks increase or decrease through a co-flow structure driven by the associated flows of the stock 'PC Units in Use'. Number of PC units in use increase through 'New PC Shipments', which is based on historical data merely data driven, and decreases through 'PC Obsolescence'.

'New PC Shipments' invoke new OS installations. The fraction of new PCs on which MS Windows or GNU/Linux are installed is influenced by the relative breadth of applications run on each OS, and the relative number of PC vendors that bundle MS Windows only or both OSs on their newly shipped PC units.

As PC units become obsolete and go out of use, the OSs that run on those PC units also go out of use, thus the outflows that decrease the stocks of 'MS Windows in Use' and 'GNU/Linux in Use', namely 'MS Windows Phase Out Rate' and 'GNU/Linux Phase Out Rate' are calculated as functions of 'PC Obsolescence', and relative market shares of respective OSs.

Another structure that influences the number of MS Windows and GNU/Linux in Use is 'Fractional Change from Windows to GNU/Linux', which is determined by relative breadth of existing MS Windows and GNU/Linux applications, and a number of scenario elements such as reliability, user friendliness, price and distribution concerns. Under reliability the stability and robustness of the OS are subsumed, user friendliness refers to the ease of use e.g. given by a graphical user interface (GUI), while distribution concerns reflect the ease of distribution and redistribution of applications. Proprietary applications may not be distributed, while open source / free applications may.

The feedback loops in this sector do not significantly influence the overall model behavior. However, certain paths in this sector are parts of larger multi-sector feedback loops that govern the general behavior of the model. Relative market shares of OSs influence the application developers willingness towards building applications for these OSs, which in turn effects relative breadth of applications run on each OSs. Relative breadth of applications influence the numbers of MS Windows and GNU/Linux in use through the fractions of new suites and the fractional change between two OSs. Four main positive feedback loops formed by these effects are shown in Figure 4.



Linux in Use
MS Windows in Use
New MS Windows Suites Fraction
Relative Market Share of MS Windows
Fractional Change from Windows to Linux
Relative Breadth of Relevant MS Windows Applications
Relative Application Developer Willingness towards Linux
+
-

Figure 4. Multi-sector feedback loops involving paths from OS Customers Sector

II. B. 4. Market Effects Sector

The effect of relative market share of two OSs on application developer willingness to build applications for either OS is calculated in this sector. This is done by calculating the direct effect of market share on developer willingness to develop for or on any given platform, and the effect of Microsoft's domination efforts on developers willingness to develop on a platform which is also determined by relative market shares.

III. Model Behavior

III. A. Base Case Run

The base case of the model is run under the assumption that all the scenario elements worked perfectly for GNU/Linux, which we believe is the case today.

Figure 5 shows the behaviors of the main stocks in the model. It is observed that GNU/Linux OS Developers increase exponentially, while MS OS Developer Employees increase linearly as intended. The sheer difference between the two OS developer stocks begins to work in favor of GNU/Linux after the 12th quarter, and GNU/Linux OS Capability, showing an exponential growth itself, surpasses MS Windows OS Capability in the 29th quarter.

The increase in GNU/Linux OS Capability affects the distribution of application developers in favor of GNU/Linux, and as the number of GNU/Linux Application Developers reaches almost half of the number of Windows Application Developers, the GNU/Linux Application Pool reaches three fourths of the Windows Application Pool. The two main reasons why GNU/Linux enjoys a relatively high number in the application stock compared to the low number of GNU/Linux application developers are that GNU/Linux application developers' productivity is higher, and the life span of GNU/Linux applications is longer.

The dynamics in the Application Developers Sector soon show their effect on the OS Customers Sector. The increase in the number of MS Windows OS suites in use begins to slow after the 28th quarter, and by the 33rd quarter this number even begins to decrease. Meanwhile GNU/Linux OS suites show an exponential growth behavior, as expected.

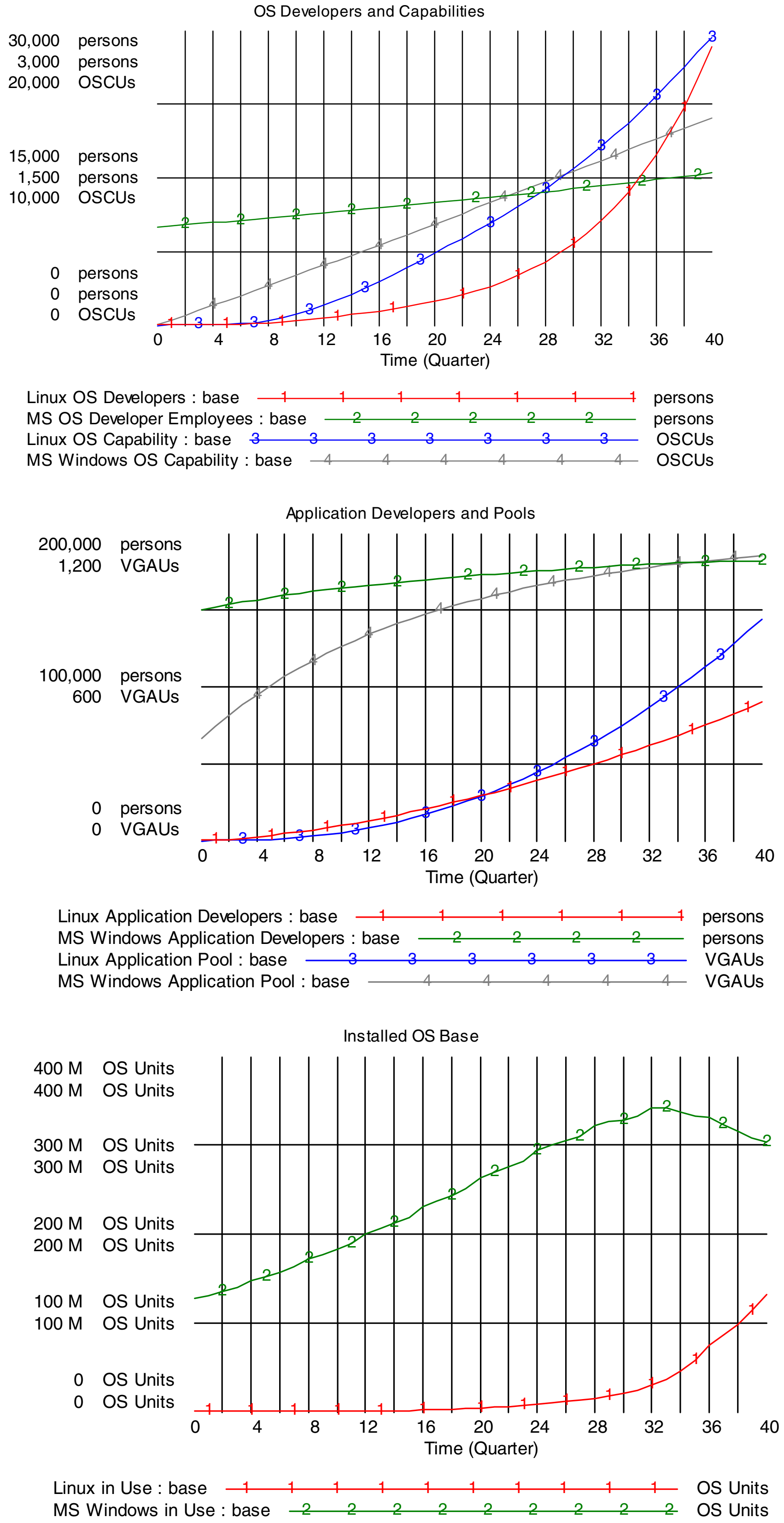


Figure 5. Output from Base run of the model.

III. B. Validation Runs

III. B. 1. Extreme Condition Tests

A number of extreme condition runs were performed to test model's robustness. As a part of these test, both GNU/Linux OS Developers and GNU/Linux OS Capability were set to zero, representing a case where GNU/Linux OS does not exist. The stocks 'GNU/Linux Application Developers', 'GNU/Linux Application Pool', and 'GNU/Linux in Use' stays at zero, as expected.

As another extreme condition test all scenario variables are set to zero, representing the case that GNU/Linux has no leverage points, such as code sharing over the Internet, open source programming, etc., at all. As expected, both OS and application developers of GNU/Linux hardly increase, as seen in Figure 6, showing that the OS stays more of a 'cult thing' which is of interest only a small group of closely related geeks and hackers. Accordingly, GNU/Linux OS capability and application pool almost stay zero. The very slight increases observed in Figure 7 are attributable to the 'hobby level' efforts of this small group of programmers. As expected the number of GNU/Linux OS suites in use remains at zero.

III. B. 2. Sensitivity Analyses

Sensitivity runs are another set of validation tests that are performed in order to test a model's robustness. One of these sensitivity runs involves 'GNU/Linux Application Developers Productivity'. 200 runs are made for values between 75% of the original value, and 125% of the original value. As it can be observed in Figure 8, the model is somewhat sensitive to changes in the value of 'GNU/Linux Application Developers Productivity', which suggests that this parameter should be further investigated.

Another sensitivity run involved 'Weight of Scenario Elements Effect on New GNU/Linux OS Developers'. 200 runs were made for values between a lower bound of 50% of the original value, and an upper bound of 150% of the original value. The graph for GNU/Linux OS Developers in Figure 9 shows that the model is sensitive to changes in this variable. However that effect does not heavily impact the number of GNU/Linux OS suites in use as can be seen in Figure 9.

III. B. 3. Scenario Tests

As the next step in the validation procedure, some scenario tests were performed. One illustrative example for these tests is where Communication and Philosophy elements are set to 0.05, 5% of the original value. This represents the case when fast and inexpensive communication media such as the Internet can be used only in a very limited manner, and philosophical elements such as open source software and code sharing are not playing any major role as in the real world. It can be observed in Figure 10, that increases in key stocks related to GNU/Linux are far from being as dramatic as in the base. Even though the numbers of GNU/Linux OS suites in use increase considerably after the 28th quarter. GNU/Linux does not seem to be a real threat to MS Windows within the selected time horizon. We consider 40 quarters to be a reasonably long period for analyzing this problem in the PC/server software market.

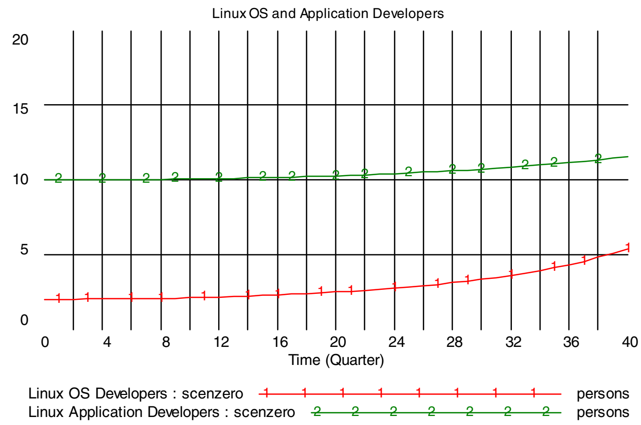


Figure 6. Output from extreme condition run: all scenario elements set to zero.

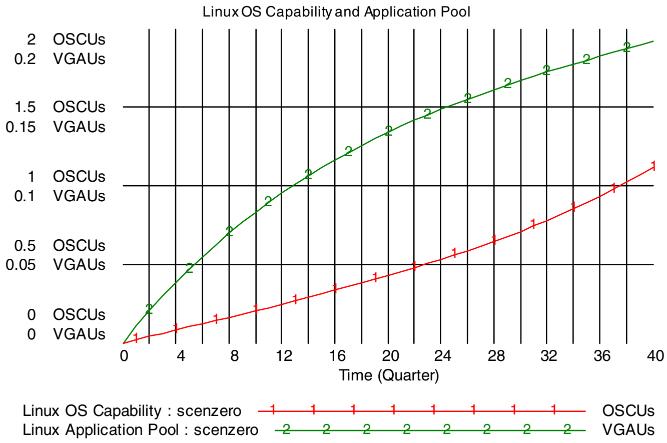


Figure 7. Output from extreme condition run: all scenario elements set to zero.

IV. Conclusions

We better understand the nervous public reactions of Microsoft executives toward GNU/Linux in recent weeks. The free software movement seems to have the potential to alter at least some rules of the game in the PC and server markets. We see prominent companies opening and freeing parts of their proprietary software, and even some diehards of the proprietary movement such as Apple released key technologies (e.g. Quicktime) to the public. We cannot, however, predict from this model whether or not the new, leaner business model has the potential to redefine the entire game. For answering this question the implications and dynamics of the diverging business models need to be included into the simulation model. This might be a consequent and natural step of further investigation. We are, however, confident that the free software movement will gain even more momentum than can be discovered today. This by itself is a dramatic development given the monopolistic scenario at the outset. We suggest that the DOJ trial against Microsoft and the proposed break-up of the company - as some Microsoft critiques demand - should be given second thoughts. So far, the arguments for cracking down the company were based on (1) the conjecture of a *market failure* in the PC and server markets (with Microsoft having more than 90 percent market share), and (2) the evidence of Microsoft's monopolistic misbehavior and unfair practices against other competitors. If the potential of developments as shown in our model exists under such adverse circumstances, we believe the market failure argument cannot be maintained.

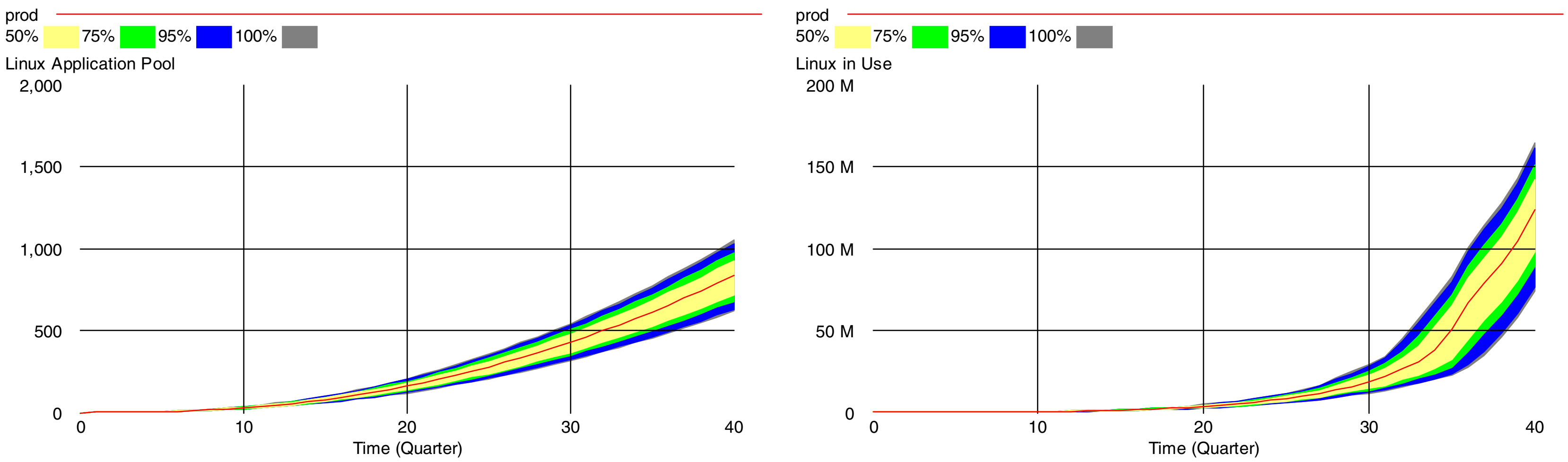


Figure 8. Output from sensitivity run: 75%-125% of GNU/Linux Application Developers Productivity.

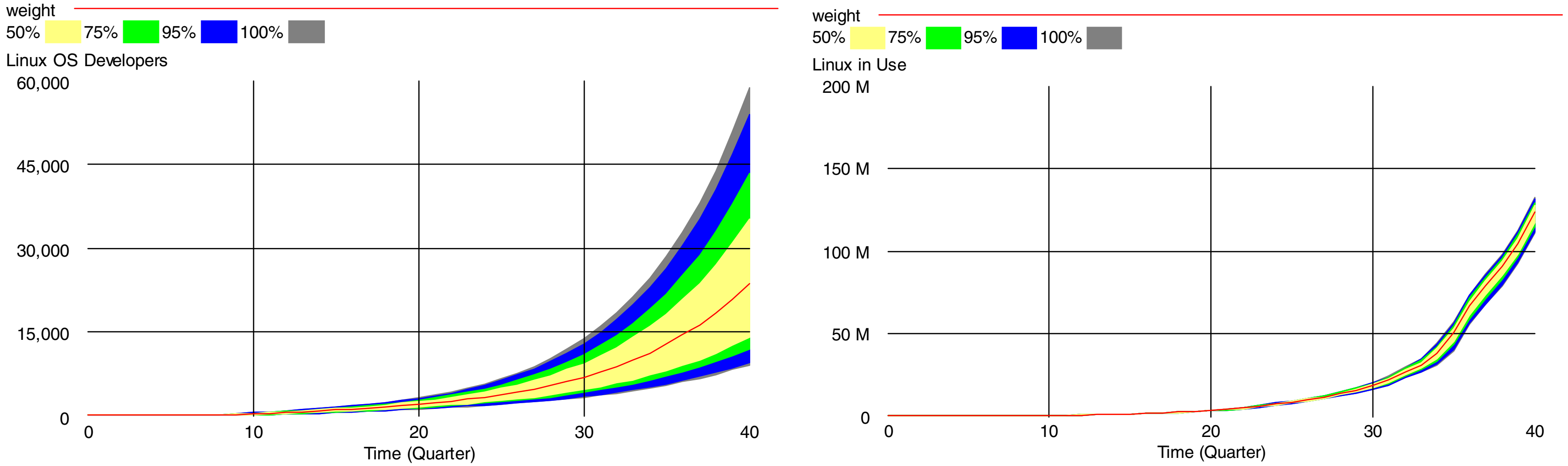


Figure 9. Output from sensitivity run: 50%-150% of Weight of Scenario Elements Effect on New GNU/Linux OS Developers.

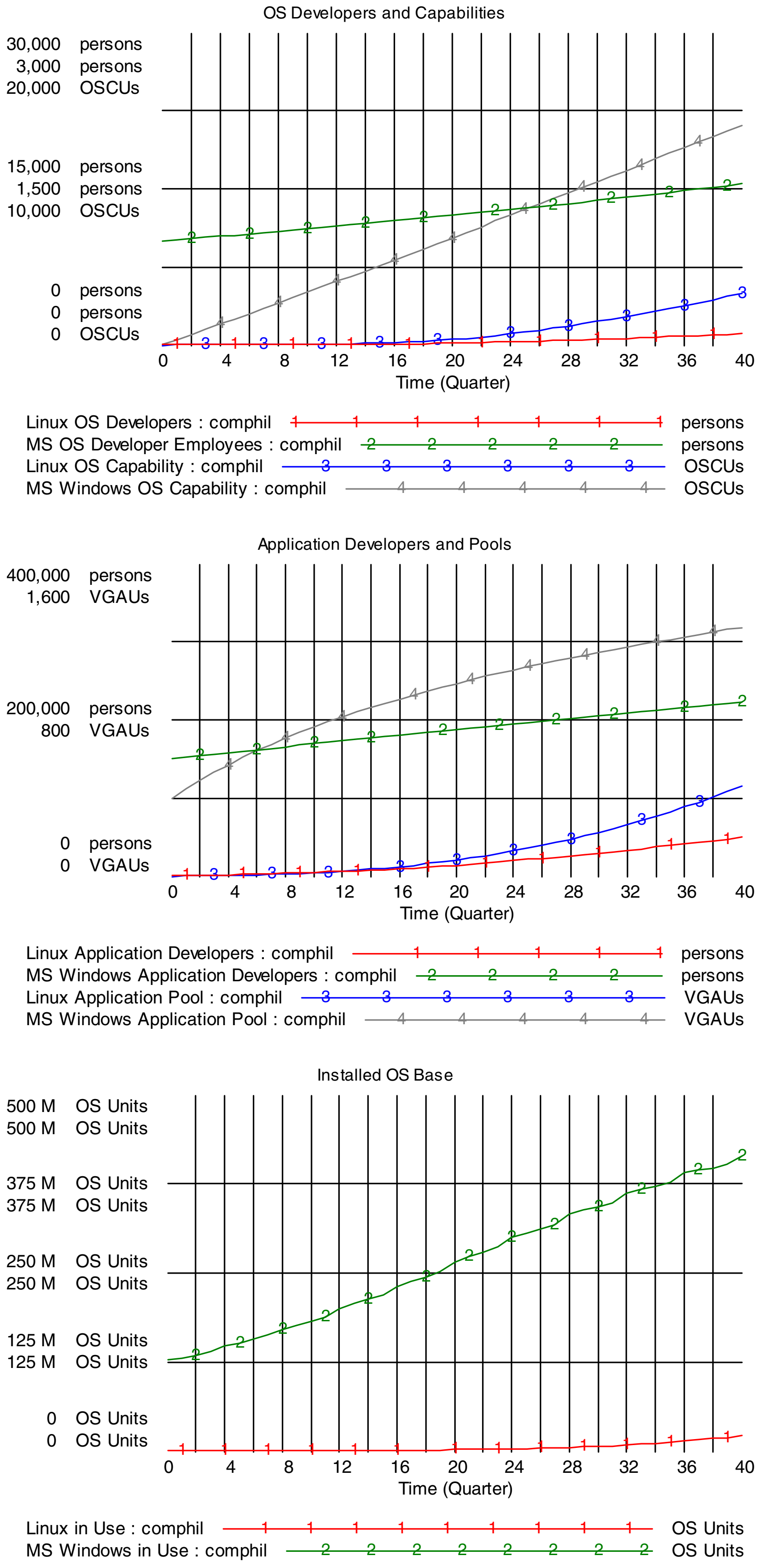


Figure 10. Output from scenario test run: Communication and Philosophy elements set to 5% of original values.