7-2003

Sell First, Fix Later: Impact of Patching on Software Quality

Ashish Arora
Carnegie Mellon University

Jonathan P. Caulkins
Carnegie Mellon University

Rahul Telang
Carnegie Mellon University

Sell First, Fix Later: Impact of Patching on Software Quality

Ashish Arora *Jonathan P. Caulkins* *Rahul Telang*

{*ashish; caulkins; rtelang*}*@andrew.cmu.edu*

H. John Heinz III School of Public Policy and Management

Carnegie Mellon University

Abstract

We present an economic model of fixing or patching a software problem after the product has been released in the market. Specifically, we model a software firm's trade-off in releasing a buggy product early and investments in fixing it later. We first show that patching investments and time to enter the market are strategic complements such that higher investments in patching capability allow the firm to enter the market earlier. Just as the marginal cost of producing software can be effectively zero, so can be the marginal cost of repairing multiple copies of defective software by issuing patches. We show that due to the fixed cost nature of investments in patching, a vendor has incentives to release a *buggier* product early and patch it later in a larger market. We contrast this result with other physical good markets. Thus, we show that a monopolist releases a product with *fewer* bugs but later than what is socially optimal. We extend our model to incorporate duopoly competition and show that in competition, the high value firm always enters earlier than the monopolist. Ironically the firm offering greater value to customers releases a product that initially is of lower quality (more bugs), but provides the greater value by releasing early (so customers can use the product sooner) and by investing more in patching so it can provide better after-sale support to its customers.

July 2003

1. Introduction

Software has become an intrinsic part of the very fabric of our lives. Virtually every commercial as well as government and non-profit enterprise depends on software to provide production, operations and marketing services to its customers. In 2002, the U.S. software market was worth almost $180 billion, with 697,000 employed as software engineers and an additional 585,000 as computer programmers (NIST, 2002: 1).

As software becomes ingrained in daily business activities, its failures become even more critical. Media reports of catastrophic effects of software failure abound. In addition to well-known failures such as the buffer overflow bug responsible for the Ariane 5 rocket blowing up 40 seconds after liftoff in 1996 (estimated loss of over $500 million), there are less widely known instances as well. For example, a software failure interrupted the New York Mercantile Exchange and telephone service to several east coast cities (Washington Post, 1998). AT&T lost significant revenues due to faulty software in one of its CISCO switches (Business week, 1999).[1]

Although quantifying the economic cost of faulty software is an open empirical question, estimates range in the tens of billions of dollars every year.[2] For instance, a recent study suggests that even 1% downtime in one small hospital translates into $1.5 million of economic cost (Anderson, 2002). The business community pays close attention to software malfunctions. Fully 97% of the 800 managers surveyed reported software flaws in their systems in the past year. More than 90% blamed faulty software for lost revenue or higher costs. Some 62% said they believed the software industry was doing a bad job of producing bug-free software (Information

[1] More examples, some possibly apocryphal, can be found at http://www.softwareqatest.com/index.html
[2] A recent NIST report claims that major software bugs cost the US economy over $60 billion per year.

Week, 2002). Software quality has become even more important recently due to heightened security concerns arising from software vulnerabilities.[3]

The final quality experienced by a software user is a combination of the bugs or defects in a software product when it is shipped plus the efforts the vendor makes to fix the bugs after shipping. Indeed, most models of the software development cycle explicitly incorporate the possibility of improving the product after release (Evans and Reddy 2002). In this paper, we analyze how a software vendor decides to allocate effort along these two dimensions. Therefore, the software vendor in our model can delay product release to reduce bugs or invest in ex-post (after the product is shipped) remediation of the bugs. There is ample evidence that many leading software vendors routinely release patches, at least in certain application domains. Sun released more than 200 patches for Solaris 9.0 to fix more than 1200 bugs in a six-month period (downloaded from http://sunsolve.sun.com). Similarly, Microsoft issues patches on an almost daily basis (The Economic Times, 2002), and Hewlett-Packard puts out an average of 60 to 80 HP-UX patches per week (InterWorks, 2000).

We assume that bugs can be reduced only by delaying the release of the software product. It is a commonplace about software vendors that "time to market" pressures are an important reason, if not the most important reason, why software has bugs. In our model, delay is costly because, all else equal, users would rather get a product sooner so they can use it longer. It is tempting to assume that firms can reduce the time to market without affecting the

[3] The shutting down of EBay and Yahoo! websites and the Code Red virus affecting more than 300,000 computers due of vulnerabilities in Microsoft products are just some of the instances which have appeared in popular press. As software products get integrated and more organizations use networks for critical business operations, security vulnerabilities become serious threats. Microsoft has invested more than $100 million to improve the security of its products. It is retraining more than 8500 programmers to improve the security of its products (Standage, 2002).

product quality by simply increasing the number of developers.[4] However, in software, this assumption is questionable. Indeed, Brooks' famous study of the IBM OS/360 suggests the opposite: The greater costs of coordinating among more developers may outweigh any gain in efficiency, so that adding people to a project may actually delay it further without any improvement in quality (Brooks, 1995). Further, as Amdahl's law implies, there may be a natural limit to the size of the product development team, especially for new software products. In fact, industry leaders have recognized that development teams should be kelp small, constant and manageable for optimal performance (The Wall Street Journal, 1991).

To capture this notion, and to focus on the time-to-market pressures, in our model, a vendor can reduce the number of bugs, albeit at a diminishing rate, only by delaying release. In addition, the vendor commits to an investment in patching that may alleviate some of the disutility due to bugs. The higher this investment is, the greater a user's willingness to pay for the product.

1.1 Research Question

We formally analyze how the ability of a vendor to patch the bugs later affects its decision about when to enter the market, and how buggy the product is. We analyze these choices under different market structure; i.e. monopoly and duopoly, and compare to a socially efficient benchmark. Finally, we note how these decisions change when the product is a tangible good like an automobile instead of software.

[4] Organizations can, and do, reduce bugs by adopting processes such as CMM or ISO9001. Ultimately, however, having fewer bugs at the time of release depend on careful design, coding and extensive testing under different conditions and for different tasks. This takes time.

A key finding is that the ability to fix bugs later always leads to an earlier release of the product. But a more striking finding is that it is indeed socially optimal to release the product early and to patch extensively later. In other words, in market equilibrium, a monopolist actually releases products later and with fewer bugs than is socially optimal. But the monopolist is also less willing to patch as extensively. Interestingly, a larger market increases incentive to release the product earlier with more bugs. Competition does cause a firm to release earlier, but still later than is socially efficient.

Ours is a full information model where users are fully aware of the quality as well as the producer's ability to fix it later. Hence, any sub-optimality is not due to incomplete information. Further, since a vendor is assumed to be able to commit to patching, we ignore uncertainty. Moreover, since the investment in patches is made (or committed to) before the user buys the products, our model is valid even if the number of bugs is stochastic.

It is important to ask whether there is something special about software that may lead to systematic market failure. There are indeed important differences between software and typical physical products. The ability to fix or otherwise mitigate software defects after release is an under-appreciated difference and is the focus here. Physical goods can of course be recalled to fix defects. However, the key distinction is that while the cost of fixing a software bug ex post is roughly independent of the number of copies of the software already sold, recalling and fixing an auto model to fix a quality problem can be very expensive, and more pertinently, will vary directly with the number of cars of that model that have been sold.

In contrast, software vendors systematically rely on being able to fix bugs later. Although the cost of actually identifying a bug and writing and testing the code to fix it can be significant, it often has a fixed-cost nature *in the sense of being independent of the number of* *copies sold*, particularly when patches can be distributed via the Internet. Of course, sometimes patching costs may vary by number of users (copies sold). For instance, if the software vendor sent a consultant to each user's site to help install the patch, then the marginal cost per user would also be non-negligible like remediation of Y2K problem required programmers to go through each organization's systems to fix code. It may also be true for some custom-developed or very complex software. However, a more common practice is to email the patch to customers or simply make it available on the vendor's website with no customer specific adaptation or hand-holding. So, to caricature, in our model, the software aftermarket repairs entail only fixed costs; for physical goods, there are both fixed and variable costs.

The fixed cost nature of patching investment implies that the effective market size plays a crucial role. Indeed, we formally show that all else equal, increase in market size implies earlier shipping (longer "time in market") but also more investment in patching (after-sale support), and higher overall value to the user. This result provides intuition for the otherwise unexpected result that, relative to the socially efficient outcome, a monopolist ships software products later, with fewer bugs, but commits to less patching later. In contrast, we show that for physical products market size is irrelevant; both the monopolist and social planner provide the same levels of defects and patch post sale remediation. We show that these results arise due to interaction between differences in market coverage, the fixed cost nature of patching, and strategic complementarity between the patching technology and the "time in market".

We extend the model to duopoly competition. First, we show that competition forces both firms to differentiate their products and offer different times of entry and levels of patch support. Interestingly, we find that when patching is a viable alternative, competition forces the high value firm to enter the market early with buggier products. But it provides higher value by investing heavily in ex-post remediation. This incentive increases with market size. On the other hand, a low value firm tends to provide little or no after sales support. Eventually, if patching is viable for both firms, both will enter a larger market earlier.

The paper is organized as follows. We provide a literature review in section 2. We outline the model, including the monopolist's and social planner's decisions, in section 3. In section 4, we provide the comparative analysis with a tangible good. We extend our model to incorporate competition in section 5. Finally, we present managerial insights and concluding remarks in section 6.

2. Prior Literature

Much work on software quality and reliability has been done from a computer science or software engineering point of view. The focus has been on process models of software development (e.g., Buckly 1984, CMM; Capability Maturity Model). It has been argued that better process leads to higher quality and lower costs and maintenance (Krishnan et al. 2000; Banker, Davis and Slaughter 1998). In fact, CMM has become a de-facto industry standard to improve the processes that lead to high quality software production. There is also unanimity that catching a bug late in the software development cycle is costly and that maintenance activities are a significant part of total cost over the software life cycle (Arthur, 1988). Firms are also investing in new testing and quality assurance programs to improve the quality of their software (Buckley 1984; Abdel-Hamid 1988). However, as noted in the introduction, our focus is not on process economies; we assume that firms are using the best available practices to produce their product. We are more interested here in analyzing the trade-off between delaying product release to reduce the bugs and entering the market early.

Cohen et al (1996) analyze the problem of performance and time-to-market tradeoff for physical products when repairing defects after product release is exorbitantly costly. Moreover, unlike this paper, they do not analyze the impact of competition nor provide comparisons to the social optimum. The literature shows that higher performance provides higher value to customers (Zirger and Maidiqui 1990; Dyson 1991), but it also leads to significant development delay (Griffin 1992; Yoon and Lilien 1985). Cohen and Whang (1997) provide a model for products and after-sale support. But in their model, after-sale service is independently sold to users, and therefore, an independent source of profit for the vendor. Our analysis differs from much of the work to date in that we allow the vendors to issue patches after the product has been released, as the empirical evidence overwhelmingly shows is done for software (Viega and McGraw 2001). In marketing parlance firms adopt a "penetrate-and-patch" strategy. Moreover, patching has become so common that many organizations have "patch-management" systems in place to efficiently integrate patches in existing software (InterWorks 2000).[5]

We also draw from product differentiation models of economics. We build on the models of vertical quality differentiation, (e.g., Mussa and Rosen 1978; Shaked and Sutton 1983; Moorthy 1983; Ronnen 1991) although unlike these models, overall quality (or value, as we call it) depends on the interplay between time of product release (which determines the number of bugs) and patching intensity of the producer.

3. Model

3.1 Consumer's Utility

All else equal users prefer software products that are released earlier and have fewer defects. In other words, a product will become obsolete on a given date, then sooner the

[5] See www.usenix.org/publications/library/proceedings for recent papers on patching and patch management.

customers get the product, the longer they can use it and the more utility they get. We define user utility as follows:

$$U = \theta\,[V - k(\delta)B(t)]\,t - p \tag{1}$$

V is the baseline utility per unit of time if the product is bug free, $B(t)$ is the number of bugs when the product is released, and t is the amount of time a user uses the product, where for simplicity we ignore time discounting. Time t is measured backwards, so a high t means the product is released early giving the user more time to derive utility from the product while lower t means that the product is released late. Note that the number of bugs, $B(t)$, is modeled as being driven by the speed with which the product is released to the market. In particular, rushing to release results in more bugs, and at an increasing rate, so we assume that $B(t)$ is increasing and convex in t, i.e., $B' > 0$ and $B'' > 0$.

The price paid by the customer is p, and the coefficient θ determines how much utility a customer derives per unit time from the software's functionality. Higher θ users derive more utility from the product, but they also suffer higher disutility due to failures. Thus, frequent users derive more utility but also incur higher loss if it fails. For analytical tractability, we assume that θ is uniformly distributed between [0, 1].

A key parameter of the model is δ which captures the firm's investment in after sales patching support, and $k(\delta)$ is the proportion of defects costs (to the user) reduced by the firm through remediation of one sort or another after the product is released[6]. Since $k'(\delta) < 0$ and $k''(\delta) > 0$, the utility loss decreases as patching investment increases, albeit at a diminishing

[6] One easy interpretation of $k(\delta)$ is that $k(\delta) = [1 - g(\delta)]^{*}D$ where D is the disutility per bug and $g(\delta)$ is the proportion of the potential disutility due to bugs that is ameliorated by the investment in patching d. Naturally, $g(\delta)$ will be increasing and concave. Therefore, $k(\delta)$ is decreasing and convex or $k'(\delta) <0$ and $k''(\delta) > 0$.

rate. The proportion of defects costs reduced, $k(\delta)$, ought to be interpreted as an average over time or in an expectation sense. For some products, a proportion of defects might be fixed immediately via a "patch" available for download shortly after the purchase. For some, $k(\delta)$ could be interpreted as the proportion of the cost of the defect defrayed by good help desk support. For others, the proportion of defects remediated might initially be zero, but might increase over time so that on average, over the time period t during which the product is consumed, that portion of defects' costs remediated is $k(\delta)$. One can possibly think of at least three distinct types of costs of defects to users:

(i) Ongoing degradation in performance because the patch is an imperfect remedy relative to the product having been designed right in the first place,[7]

(ii) The cost of "failures" that occur between the time the defect is discovered and when it is fixed or patched, and

(iii) The cost to the user of installing the "patch" (e.g., the cost of redoing systems integration testing when one software component is updated or patched).

So our model focuses on first category and the second category if the time between defect detection and repair looks more like a proportion of the total time the product is used than a fixed constant. The third category could be accommodated by adding a negative fixed cost per patch.

3.2 Vendor's Profit Function

A general formulation of the software vendor's objective function is

$$\pi = Q(p - f\delta B(t)) - G(\delta, B(t), t) \tag{2}$$

[7] One example would be Microsoft Outlook's vulnerability to email viruses. The basic problem is "built in" to the architecture and cannot be fixed even now that it is recognized. Hence, users need to frequently update their virus protection software. Microsoft's having a large team of people who make virus updates available very quickly would be like having a large δ. Another example would be if the patch just lets the system "crash" more gracefully, but does not reduce the frequency of crashes. The cost of the bug is partially defrayed ($k(\delta) < 1$), but the bug continues to reduce the utility provided by the software even after it is detected and patched.

where Q is the volume sold, p is the price, and the other terms represent costs of patching. As we noted in the introduction that firms can only reduce the bugs via delay, we can assume without the loss of generality that initial product development costs are fixed costs and do not affect the firm's optimization problem.

In general, patches can involve both fixed and variable costs to the vendor. The constant f represents the per unit *variable* cost per defect remediated. The second term in (2), $G(\delta, B(t), t)$ is the *fixed* (relative to volume of sales, Q) cost of remediating defects. This term would include, for example, the cost of doing the research necessary to figure out how to remediate a given defect. For an auto manufacturer, this would be the cost of designing a substitute for a defective part. For a software vendor, it could be the cost of writing the code for a "patch". Note that these costs vary in that vendors may choose to patch defects more promptly, requiring greater investment up front.[8] They may also vary with the number of defects. What matters is that they do not vary by the number of users, Q.

One focus of the analysis will be contrasting "tangible" products, for which f is large, and "software" products for which f is smaller, if not negligible. To make the contrast between tangible and software products as stark as possible, and for analytical convenience, for the "tangible" products case ($f > 0$) we will assume that the fixed costs of defect remediation are negligible ($G() = 0$) and for software products that f is literally zero and the fixed costs are significant. In reality, even downloading patches from the vendor's web site imposes some (trivial) marginal cost on the vendor in the form of server utilization. But the polar cases are stylized renderings of the typical conditions for manufacturing and software development.

[8] In this sense, investments in patching are an instance of an endogenous fixed cost. Sutton (1984) provides a more general analysis and implications for market structure.

At this point, to continue the analysis, it is necessary to get more specific about how $G(.)$ depends on its arguments. One vision of defect remediation is that for every defect a solution must be invented. Thus, patch development costs are proportional to the number of patches.[9] Therefore, we choose the cost function $G(.) = FC(\delta)B(t)$, with $C(\delta)$ increasing and convex in δ. One can see that $G(.)$ is increasing in δ and t, at an increasing rate. Our results are robust to other plausible formulations.[10]

3.3 Market Equilibrium under Monopoly

Given Equation (1) for the users' utility and the uniform distribution of θ, demand for the software product facing a monopolist vendor is

$$D(p,t,\delta) = \left(1 - \frac{p}{[V - k(\delta)B(t)]\,t}\right).$$

Hence the profit Equation (2) for the monopolist profit becomes

$$\pi(p,t,\delta) = \left(1 - \frac{p}{[V - k(\delta)B(t)]\,t}\right)p - FB(t)C(\delta) \quad . \tag{3a}$$

It can be readily verified that after substituting for the profit maximizing price,

$$\pi(t,\delta) = \frac{1}{4}[V - k(\delta)B(t)]\,t - FB(t)C(\delta) \quad . \tag{3b}$$

[9] Many large firms respond to the bugs in their products by releasing appropriate patches. For example, CISCO specifically responds to all the bugs discovered (http://www.cisco.com/warp/public/707/advisory.html#notices).

[10] Alternate models would view the vendor's commitment in terms of effort budgeted for responding to defects: the vendor may assign γ programmers to develop patches. Hence, the cost to the vendor is F γ t, where F is the cost per programmer per unit time. The average proportion over time of defect costs remediated is presumably an increasing, concave function of γ, i.e., $\delta = f(\gamma)$ with f' > 0 and f'' < 0. So, expressing the model in terms of δ, the cost function becomes F c(δ) t, where c' > 0 and c'' > 0. Alternately, the budget for developing patches could be set simply in terms of the total programming effort. In that case, the cost to the vendor would be just $F\gamma$, where F is the cost per programmer-year. Our qualitative results seem robust with respect to these variations in the cost function, but other variations may prove fodder for extensions of this paper's analysis.

It is useful at this stage to introduce $\Psi(\delta, t; m) = m[V - k(\delta)B(t)]\,t - FB(t)C(\delta)$. We assume that $\Psi(\delta, t; m)$ is concave in t and δ. This assumption is required for an interior maximum. Concavity is hard to show for the general functional forms we used, but is readily satisfied for various specific common functions. (See Appendix.)

The first order conditions for t and δ are

$$\frac{d\pi(t,\delta)}{dt} = \frac{1}{4}[V - k(\delta)B'(t)\,t - k(\delta)B(t)] - FB'(t)C(\delta) = 0 \tag{4}$$

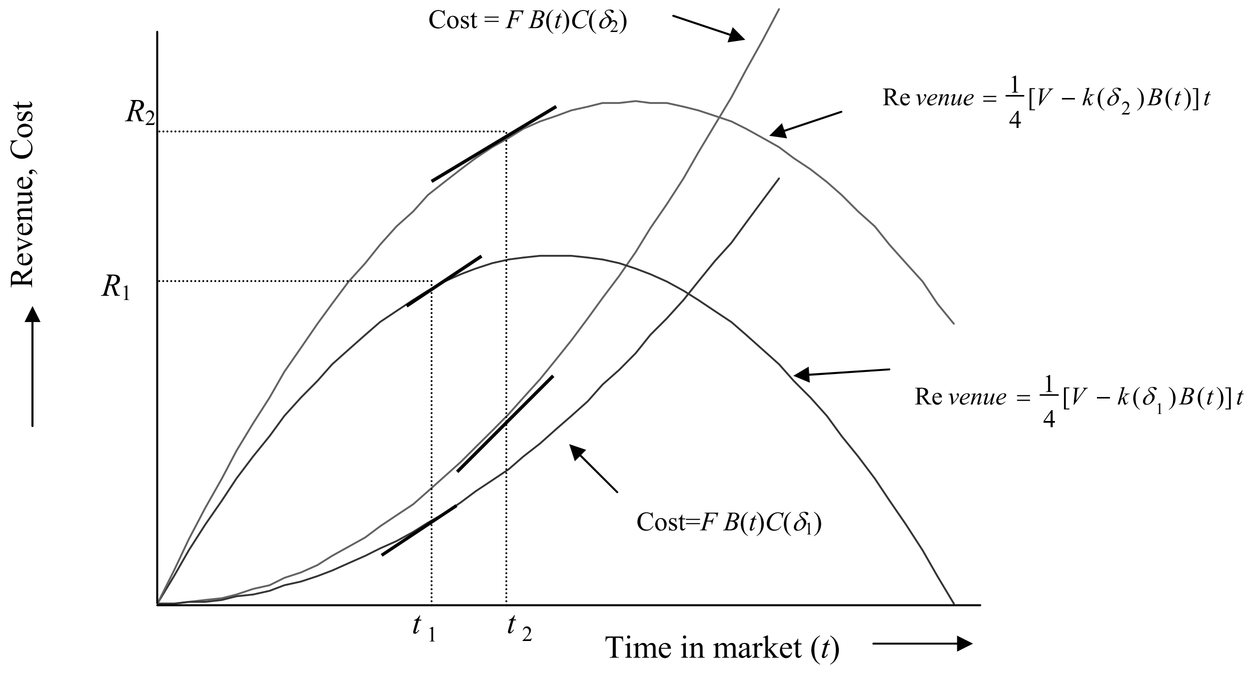
$$\frac{d\pi(t,\delta)}{d\delta} = -\frac{1}{4}k'(\delta)B(t)\,t - FB(t)C'(\delta) = 0 \ . \tag{5}$$

For future reference, note that for δ satisfying the first order conditions

$$\frac{d^2\pi(t,\delta)}{d\delta\, dt} = -\frac{1}{4}k'(\delta)(B'(t)\,t + B(t)) - FB'(t)C'(\delta) > 0 \tag{6}$$

since $-\frac{1}{4}k'(\delta^*)t^* = FC'(\delta^*)$ from eq (5). Thus δ and t are "strategic complements" (Milgrom and Roberts, 1990), implying that factors that increase the marginal payoff from δ will also result in a higher value of optimal t. In other words when investing more in patching is desirable, the vendor releases the product earlier.

Figure-1 below illustrates the intuition behind this result and highlights the generality of the cost function we have used. Note that $\frac{1}{4}[V - k(\delta)B(t)]\,t$, the total revenue of the monopolist, is a concave function of t for a given δ. The costs are convex in t. A monopolist will choose an optimal pair (t,δ) which maximizes the vertical distance between the value and cost curves. As can be seen, the monopolist's optimal time in market is given by t_1 for $\delta = \delta_1$. A higher choice of δ, $\delta_2 > \delta_1$, implies that the revenue curve shifts up since patching is beneficial for users, for any given t.



Cost = $F\,B(t)C(\delta_2)$
$\text{Revenue} = \frac{1}{4}[V - k(\delta_2)B(t)]t$
R_2
R_1
Revenue, Cost
$\text{Revenue} = \frac{1}{4}[V - k(\delta_1)B(t)]t$
Cost=$F\,B(t)C(\delta_1)$
t_1
t_2
Time in market (t)

Fig-1: *Complementarity Between t and δ*

The key to our results is that a higher δ also increases the marginal payoff of t, so that the revenue curve for δ_2 has a higher slope (algebraically) for all values of t. Moreover, although increasing δ also increases cost, the increase in marginal cost is not as great as the increase in marginal revenue. As a result, the optimal t^* increases with δ.

This figure also provides insight into the implications of different forms of a cost function. For instance, if the cost function were additively separable in $B(t)$ and $C(\delta)$, then changes in δ would imply a parallel shift in the cost curve, leaving the marginal cost unchanged. It follows that this would leave the complementarity between t and δ intact, and our results would be true *a fortiori*. Similarly, if costs were independent of $B(t)$, then the marginal cost with respect to t would be zero so that the cost curve would be parallel to the horizontal axis. In this case, the optimal t is simply given by the point at which the value curve attains its maximum. Again, it is straightforward to see that t and δ are complementary, so that increases in δ are accompanied by increases in t. Thus, the multiplicative cost function we have used is quite general and our qualitative results robust to other plausible formulations.

3.4 Benchmark: The Socially Efficient Outcome

It is interesting to compare the choices of the monopolist in time of entry and patching support to their socially efficient levels. Efficiency requires that the product is priced at marginal cost[11], which we have assumed here to be zero. At this price, the entire market will be covered. Therefore, the objective function to be maximized is -

$$S(t,\delta) = \int_0^1 \theta[V - k(\delta)B(t)]\,t\,d\theta - FB(t)C(\delta) = \frac{1}{2}[V - k(\delta)B(t)]t - FB(t)C(\delta) \qquad (7)$$

Note that equation (7) and the monopolist's profit function have the form $\Psi(\delta, t; m)$, where m = ½ for the social planner and m = ¼ for the monopolist. We prove in the Appendix that the optimal values δ^* and t^* that maximize $\Psi(\delta, t; m)$ are increasing in m. This implies that the socially efficient outcome is to release the product earlier than the monopolist (and hence with more bugs) but patch more aggressively, as summarized in Proposition 1 below.

Proposition 1: *For a software product, the monopolist releases the product later with fewer bugs but invests less in patching and ex-post support than the socially efficient levels.*

[11] The result holds even if we impose a "break even" constraint, since, as we explain, the key is that the monopolist serves fewer customers than is socially efficient.

The result is surprising in that contrary to popular belief, monopolist actually releases less buggy products than what is socially optimal. But monopolist provides less value by both entering the market later and by providing less ex-post support.

3.5 Role of Market Size

In our analysis, the market size is normalized to one and m captures the proportion of market captured by the monopolist or social planner. Should the market be larger, then for the same m, there will be more users buying the product. Thus, we can also interpret m as an indicator of market size. It follows from the proof of Proposition 1 that a producer facing a larger market has a greater incentive to enter the market earlier with a buggy product and provide extensive patching support later. The costs of patching can be amortized over a larger sales volume, inducing greater investment in δ. Since δ and t are strategic complements, greater investment in δ is associated with earlier (and hence buggier) product release (i.e., a larger t). This result is also clear from Figure 1. An increase in market size shifts the value curve up and to the right for a given δ. Thus, all else constant, a monopolist with a larger market would choose higher levels of δ and t. By parallel reasoning, the smaller the fixed cost per bug patched (smaller F), the greater are δ and t. Thus a corollary of Proposition 1 is:

Corollary 1: *A software producer facing a larger market or lower fixed costs per bug patched enters the market earlier with more bugs but provides higher patching support later.*

Thus, our model provides a clear and testable prediction, particularly if the number of patches is interpreted as the frequency with which newer versions are released by the vendor.[12] Our model implies that, in a large market, one should see frequent versions being released by the

[12] Generally newer versions have more features as well. In our model we ignore features. Similar to Jackson (2002), in a non-traditional definition, lack of features can be interpreted as a bug.

vendor. The costs of patching also depend on the type of product and domain in which firms operate, such as systems software, and application software. Our results suggest that when the cost of patching is small or when the vendors have better support infrastructure, it has an incentive to enter earlier with buggier products.

Microsoft typically has large market for its products, and also provides extensive patching support. Many of its patches are released in "service-packs" that make it easy for the user to download and install new patches. The users (or network administrators) do not have to worry about investing time and energy in looking for the right patches. Moreover, these service packs can also scan the computer and networks, identify bugs and vulnerabilities and automatically download the right patches. Clearly, Microsoft is willing to incur these large costs in after-sale support to reduce customer disutility. In other words, Microsoft is investing more in δ and reducing $k(\delta)$ for a user.

4. Quality in tangible products

We now compare this result with a tangible product whose producer incurs a positive marginal cost of fixing a defect. We follow the same model structure except that we replace the fixed cost per "patch" with a cost that varies with the number of units fixed, as well as with the number of bugs. Therefore, the profit function for a tangible product monopolist is

$$\pi(p,t,\delta) = \left(1 - \frac{p}{[V - k(\delta)B(t)]\,t}\right)\left[p - fB(t)C(\delta)\ \right]$$

Here $fB(t)C(\delta)$ is the marginal cost of fixing a defect[13]. Substituting for the profit maximizing price leads to the profit function:

[13] We could add a marginal cost of production 'c' as well, but without loss of generality and to focus on the economics of patching assume $c = 0$.

$$\pi(t,\delta) = \frac{\left[\left(V - k(\delta)B(t)\right)t - fB(t)C(\delta)\right]^2}{4[V - k(\delta)B(t)]\,t}. \tag{8}$$

As always, the socially efficient outcome would involve pricing the product at marginal cost. Therefore, $p = fB(t)C(\delta)$. Hence the social surplus can be written as

$$S(t,\delta) = \frac{\left[\left(V - k(\delta)B(t)\right)t - fB(t)C(\delta)\right]^2}{2[V - k(\delta)B(t)]\,t} \tag{9}$$

Since the monopolist's and the social planner's objective functions differ by only a multiplicative constant, they will make the same choices with respect to δ and *t*.

Proposition 2: *When the cost of fixing a failure ex post is a marginal cost (as in the case of a tangible good), the monopolist's choices of time to release the product and investment in patching bugs later are socially efficient.*

Why is there a difference between tangible and intangible goods? There are three important factors. First, the monopolist serves a smaller market than the social planner. Second, when patching is a fixed cost, the smaller market leads to a smaller investment in patching. Third, patching is a strategic complement to *t*. All else constant, the higher δ is, the higher is *t*. Thus, in software, the smaller market coverage of the monopolist leads to a smaller δ and therefore also a smaller *t*.

The importance of the strategic complementarity can also be seen by assuming that the patching technology is not available i.e., by setting $\delta = 0$, in the previous section's model with *fixed* costs of patching. In this case, the monopolist's profit function $\pi(t,0) = \frac{1}{4}[V - k(0)B(t)]\,t$, differs by only a multiplicative constant from the objective function for social optimality $S(t,0) = \frac{1}{2}[V - k(0)B(t)]\,t$, which leading to the following proposition.

Proposition 3: *When ex post patching of bugs is ruled out, a monopolist software vendor releases the product with the same ex-ante quality even in markets of different sizes, and it is the socially optimal quality.*

Clearly, the fixed cost character of patching software defects after product release can create a "market failure" (difference between the market equilibrium and the socially optimal outcome) that does not exist for conventional physical products, at least in the case of a monopolist. We next show that this and related findings are not restricted to the case of monopolist providers.

An alternative, but not mutually exclusive, interpretation of the difference between software and tangible goods has to do with differences in the legal environment. In many tangible goods, a product defect exposes the manufacturer to liability claims, and in general, requires restitution to the user. Typically, software vendors are not exposed to such claims. Liability can be interpreted as another way of compensating users for defects (i.e., of reducing disutility from bugs) after product release. However, a key difference with patching is that liability costs vary directly with the number of users. On the other hand liability for software developers is still different from tangible goods producers because it can be reduced after product release via interventions that have a fixed but no variable cost, namely greater investment in patching.

A priory, it is hard to say whether exposing software developers to product liability would cause them to improve the quality of products at the time of their release. The greater marginal cost per defect and per copy sold would create an incentive to delay product release in order to improve initial quality. It would also tend to reduce sales volume and, hence, the customer base over which fixed investments in patching could be amortized. On the other hand, it would increase the incentive for patching, and patching is a strategic complement with time in market. Which effects would dominate could depend on the details of how patching was viewed as reducing liability exposure, which in turn depends on the details of how one models bugs as harming users. This topic could be an interesting question for further research.

5. Competition

5.1 Preliminaries

The monopoly case is not uninteresting for software (consider Microsoft's market share in PC operating systems), but understanding the impact of competition on the timing of product release and investments in patching technology is of obvious interest as well. Here we study the case of two firms offering functionally identical software products. It is clear that the firms would like to differentiate themselves to avoid intense price competition that would otherwise erode profits. In practice, firms can and do differentiate on multiple attributes. For example, they can differentiate via different marketing efforts or by offering different product features. In our model, we allow firms to differentiate their products only with respect to the variables of interest here: product release date (ex-ante quality) and investment in patching.

The firm strategy space can be considered in two stages. In the first stage, firms announce the date of release of their product and commit to a given level of investment in patching. At the second stage, they set prices and enter the market, whereupon consumers make the rational choices given their preferences. We study the sub-game perfect equilibrium. Without loss of generality we label the firm offering higher value to customers as H [High value firm] and denote its choices by t_h, δ_h, and p_h. We denote the firm offering lower value to customers as L [Low Value firm], denoting its choices by t_l, δ_l, and p_l, so that the utility offered by the *j* type firm ($j \in$ {H, L}) to users is

$$U = \theta[V - k(\delta_j)B(t_j)]t_j - p_j$$

To derive the demand function for both firms, we presume customers chose optimally by buying from H, L, or neither. For convenience, let *q* denote the total value (gross of price) offered to the customer, so

$$q_j = [V - k(\delta_j)B(t_j)]t_j, j = H, L \tag{10}$$

Value, along with price, is sufficient for determining customer's choices and so firm differentiation can be defined first in terms of that single dimension. The users will buy from H firm when $\theta q_h - p_h \geq \theta q_l - p_l$ and buy from L as long as $\theta q_l - p_l > 0$.

Simple algebra gives the optimal prices charged by both firms (Tirole 1998; Ronnen 1991): $p_h^* = \frac{2q_h(r-1)}{4r-1}$ and $p_l^* = \frac{q_l(r-1)}{4r-1}$ where $r = \frac{q_h}{q_l}$. The revenue functions of both firms can be written as $R_H = \frac{4q_h^2(q_h - q_l)}{(4q_h - q_l)^2}$ and $R_L = \frac{q_h(q_h - q_l)q_l}{(4q_h - q_l)^2}$.

Their profit equations are

$$\pi_H = \frac{4q_h^2(q_h - q_l)}{(4q_h - q_l)^2} - FB(t_h)C(\delta_h) \tag{11a}$$

$$\pi_L = \frac{q_h(q_h - q_l)q_l}{(4q_h - q_l)^2} - FB(t_l)C(\delta_l) \tag{11b}$$

While solutions for optimal q_h and q_l are well established, solving for optimal t and δ is non-trivial. Therefore, before we establish the equilibrium, we first establish the relationship between value offered and the optimal t and δ. In other words, if a firm wants to increase the value, how will it change t and δ optimally to achieve that? We show that t and δ are (weakly) increasing in the level of value chosen.

First suppose that the desired value is low enough to be obtained with $\delta = 0$, i.e., $q \leq [V - k(0)B(\hat{t})]\hat{t} = \hat{q}$ where $\hat{t}$ maximizes the value q given $\delta = 0$. ($\hat{t}$ is well-defined since q is concave in t.) Assuming the cost of patching is zero when no patching is done (i.e., $C(0) = 0$), for some fixed $q \leq \hat{q}$, the objective function is independent of δ and minimized with respect to t. Hence, if the vendor wishes to offer a value attainable with $\delta = 0$, it will do so by manipulating t alone. In particular, since $q(t;\ \delta = 0)$ is increasing in t for all $t \leq \hat{t}$, the vendor can increase q by increasing t.

Any increase in quality beyond $\hat{q}$ will be accompanied by investments in patching. But investments in patching will affect the time to release as well. Therefore, if the firm wants to offer some value $q = \overline{q}$, then it does so in the least costly way. In other words, firms are effectively solving the following problem:

$$\begin{aligned}&\text{Min } \ F\,B(t)\,C(\delta)\\ &\text{such that } \big(V - k(\delta)B(t)\big)t = \overline{q}\end{aligned}$$

The Lagrangian function can be formulated as

$$\text{Max } L(t,\delta,\lambda) = -FB(t)C(\delta) - \lambda\big[\big(V - k(\delta)B(t)\big)\,t - \overline{q}\big]$$

The first order conditions for an interior optimum are

$$\frac{\partial L}{\partial t} = -FB'(t)C(\delta) - \lambda\big(V - k(\delta)B(t)\big) + \lambda k(\delta)B'(t)t = 0$$
$$\frac{\partial L}{\partial \delta} = -FB(t)C'(\delta) + \lambda k'(\delta)B(t)t = 0$$
$$\frac{\partial L}{\partial \lambda} = -\big(V - k(\delta)B(t)\big)t + \overline{q} = 0$$

We are particularly interested in the sign of $\frac{\partial \delta}{\partial \overline{q}}$ and $\frac{\partial t}{\partial \overline{q}}$.

Lemma 2: *At the optimal* t^* *and* δ^*, $\frac{\partial \delta}{\partial \overline{q}}$ *and* $\frac{\partial t}{\partial \overline{q}} > 0$.

Proof: See Appendix. In other words, to increase value, the firm increases its investment δ in patching which, in turn, leads to early release of the product.

The firm offers a pair (t_1^*, δ_1^*) when it wants to offer a value q_1. If the firm needs to offer another value $q_2 > q_1$, then Lemma 1 shows that it can do so most optimally by investing more in patching but releasing the product earlier and therefore offering a pair (t_2^*, δ_2^*).

5.2 Equilibrium Analysis

We show that the cost of patching in competition plays a crucial role in the patching support and release date decisions. We first characterize the equilibrium of this game. As is standard in the literature on vertically differentiated products (e.g., Shaked and Sutton, 1984; Ronnen, 1991) we focus on equilibria such that the two firms offer different quality levels: q_h^* by the high-value firm (H) and q_l^* by the low-value firm (L) such that $q_h^* > q_l^*$. From Ronnen (1991), we know that there exists an equilibrium such that q_l is non-decreasing in q_h and vice-versa.[14]

[14] Ronnen shows that an equilibrium exists as long as cost of quality Φ (q) is convex, as is the case in our model.

First note that both q_l and q_h are strictly positive. Otherwise, no customer would purchase that firm's product. Next it is easy to show that firm H always enters the market before the monopolist. Substituting $q_l = 0$ in Equation 11(a) implies that when firm L offers zero quality, firms H's objective function is identical to the monopolist's objective function (Equation 3). Intuitively, when a firm's only competitor is offering zero value, the firm is effectively a monopolist. Hence, when $q_l = 0$, firm H offers value equal to that provided by the monopolist. But we know that in equilibrium the lower value firm offers $q_l > 0$ and that q_h increases in q_l. Hence firm H always offers a higher value than the monopolist. From Lemma 1, we know that δ and t are increasing in q. It therefore follows that firm H enters earlier than it would absent competition from the low value firm and also invests more in patching. Thus, we conclude that, for the high value firm, competition shortens time to market and increases the number of bugs, but it also results in greater post sales support and patching, and overall, greater value to the consumer. We formalize this intuition in the following proposition.

Proposition 4: *In the presence of competition, the high value firm enters the market earlier than would a monopolist.*

Figure 2 displays how quality and product release times of the various firms vary with the cost of patching, captured here by variations in F. On the x-axis is the cost of patching, F. On upper y-axis is the value provided. On the lower y-axis is the time t when these firms release their products. It is easy to interpret the figure when starting from the right most side of the x-axis. When the cost of patching is greater than some threshold level, $\hat{F}$, no firm invests in patching. Firm H (broken curve) enters at time $\hat{t}$, and invests $\delta = 0$ in patching support and offers the value $\hat{q}$. Following Proposition 3, the monopolist (solid curve) does the same. But

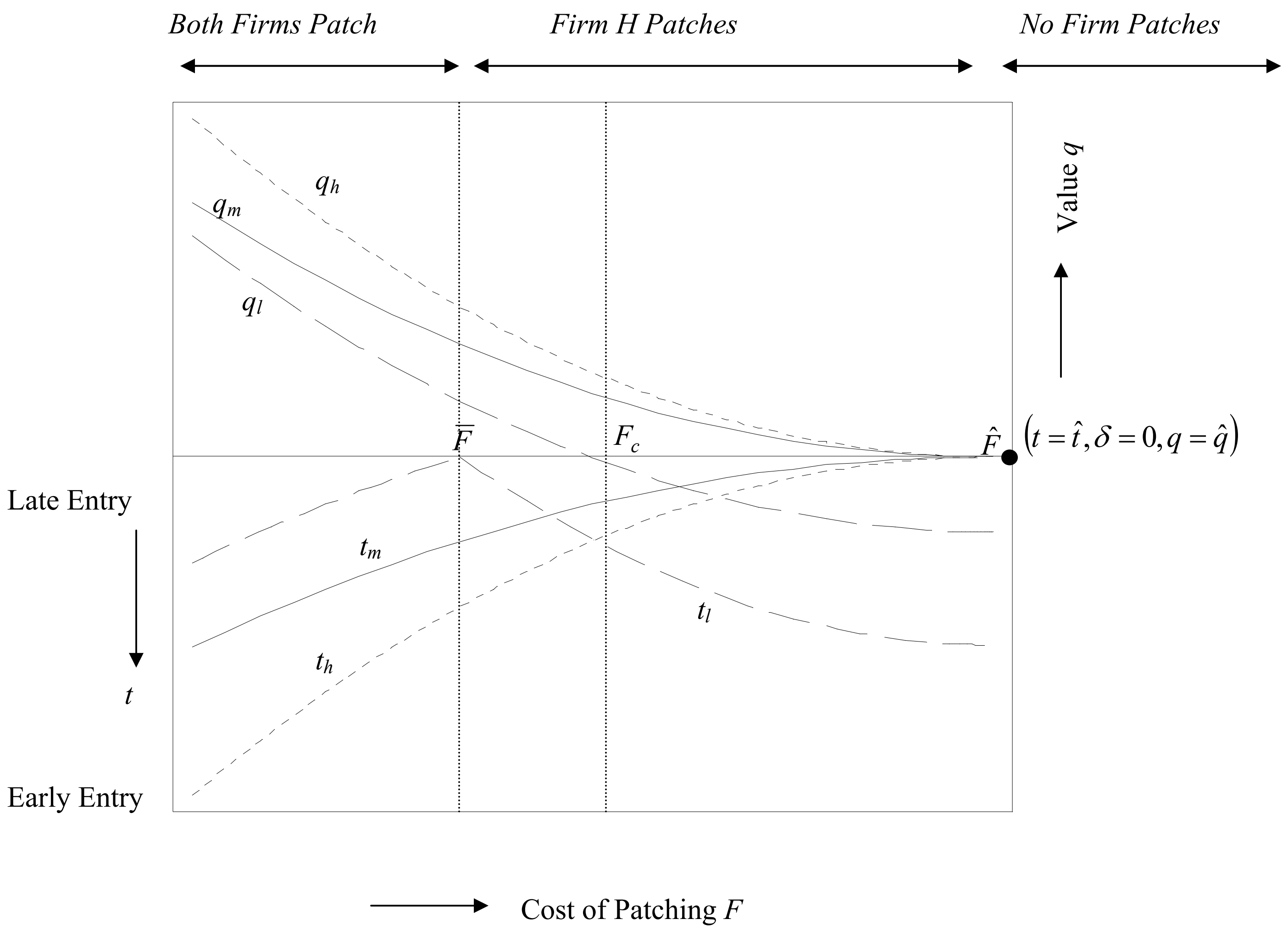


Fig 2: *Patching Cost* (δ)*, Time in market (t), and Value* (q)

firm L (light broken curve) differentiates by entering the market early and charging low prices. Clearly, firm L does not invest in patching either. Note that it offers a lower value than $\hat{q}$.

Moving to the left, as the cost of patching decreases, firm H invests in patching and enters the market earlier than the monopolist, as noted in Proposition 4. In response, firm L increases its value by entering later (but initially not patching). Eventually, when cost of patching falls to F_c , it is possible the firm H enters earlier than firm L but invests significantly more in after-sale support. For low enough costs of patching $\overline{F}$, both firms invest in after-sale patching support. As the figure also shows, regardless of how t and δ are combined, as the cost of patching falls, value provided increases.

The cost of patching will depend on factors such as the application domain and firm capabilities. For critical applications, patching may not even be an option and the cost of patching can be taken as prohibitive. In such cases, firm H will enter later than firm L and both firm H and a monopolist would produce efficient outcomes. Similarly, if the firm has the infrastructure and capabilities in ex-post support then it will produce a lower quality product and enter the market earlier, but provide better after sale support.

5.3 The Role of Market Size:

A larger market leads to more revenues for both firms without any changes in their cost structure. Therefore, in equilibrium both firms want to provide higher value. In other words, we can re-interpret Figure 2, by shifting the right vertical axis leftwards. Again, from Figure 2, it is clear that, for all else constant, a larger market will invite an earlier product release in the presence of competition. This result provides support to what is usually observed in the marketplace; i.e. competition forces a firm to enter the market earlier and patch more when the market is more attractive. Interestingly, it is firm H that has stronger incentives to enter early compared to firm L. But again, for low enough cost of patching, higher market size will lead to early entry by both firms. (See Figure 2 when cost is less than $\overline{F}$.)

While there are many reasons firms tend to enter early, including first mover advantage and switching costs, our results point to another dynamic in this industry. The ability to remediate defects after releasing the product and the fixed nature of these costs provide incentives to enter the market earlier, and the larger the market, the stronger these incentives.

6. Conclusions and Future Research

The recent focus on software quality and vulnerability makes it important to understand the incentives of software vendors to provide quality. We provide a simple economic model of the phenomenon of issuing patches and fixing software bugs by software vendors and its impact on ex-ante quality and total value provided. We analyze the firm's trade-off in time-to-release the product vs. investments in ex-post remediation of software failures. We first show that there is strategic complementarity between "time to enter" the market and "patching" investment. Therefore, as firms invest in patching, they tend to enter the market earlier with buggier product.

We also show that the fixed cost nature of investments in bug-remediation leads firms to enter a larger market early with a buggier product but also invest more in after-sale remediation. Since a monopolist restricts output relative to socially efficient levels, this leads to counter - intuitive result that a monopolist actually provides a product with fewer bugs but then provides less "after-sale" support and therefore, reduces customer value. We compare this result with a traditional tangible good case and show that when there are significant marginal costs of remediation, market size does not play any role, and thus, a monopolist's time-of-release strategy is also socially efficient. Consistent with our model, we note that that there is widespread dissatisfaction with, for instance, the many security vulnerabilities in Microsoft's products, even when compared with products from rivals with smaller market share. But few can match the extensive system of updates and patches that Microsoft provides.

We extend our model to incorporate competition involving two firms that endogenously differentiate by providing different levels of total quality or value. We show that sometimes both firms enter the market earlier than would a monopolist. But a high quality firm compensates for early entry and a "buggy" product with aggressive support for ex-post remediation. In other words, the firm provides higher after-sale support to its users. This result parallels the "penetrate-and-patch" strategy typically observed in this industry where the firms want to enter the market early to capture market share but the high value firm is willing to invest in the infrastructure to provide after-sale support. The time of entry also depends on how expensive it is for a firm to fix the bug. The easier it is to patch, the greater the incentives to release a buggier product early and patch it later. Inasmuch as the Internet reduces the cost of issuing patches, it might in this way be a contributor to the bugginess of software products.

Thus competition helps, but it still does not lead to the socially optimal levels of time of product release, reliability or bugginess, or after-market support. Furthermore, such a "market failure" in this model is unique to products, such as software, for which the marginal patching costs are zero with respect to number of users. The same model parameterized for "pure" physical products had no such market failure. The question of whether and how the government might intervene to affect software quality is extremely delicate and certainly ought to encompass many considerations besides the zero marginal cost of software defect remediation. However, this paper is certainly germane to that larger discussion.

Even within the domain of models focusing on software reliability and after market remediation, there are opportunities for extensions and further work. One could consider models of competition other than duopoly; and models in which customers are heterogeneous not just on a single dimension (θ) governing how valuable the product is to them, but also independently with respect to how costly bugs are. A further extension would allow firms to release at multiple dates, with later versions being more reliable and fewer bugs, and for customers to optimally decide on which version to buy.

Another extension would consider patching as a dynamic choice variable, not one to which firms can commit. If outcomes are worse without commitment, that could be an argument for supporting organizations like CERT, which acts as a security watch dog, and consumers consortia like the "bugtraq" mailing list, that are strong enough to force even a firm like CISCO to respond to bug queries promptly.

Appendix

<u>Proof of Concavity of profit function in *t* and δ</u>

Consider the profit function $\pi(t,\delta) = \mathrm{m}[V - k(\delta)B(t)]\,t - FB(t)C(\delta)$. The first order conditions for *t* and δ are

$$\frac{d\pi(t,\delta)}{dt} = m[V - k(\delta)B'(t)\,t - k(\delta)B(t)] - FB'(t)C(\delta) = 0$$

$$\frac{d\pi(t,\delta)}{d\delta} = -\,mk'(\delta)B(t)\,t - FB(t)C'(\delta) = 0 \quad \text{Or,} \quad -\,mk'(\delta)t = FC'(\delta)\,.$$

The second order partial derivatives are

$$\frac{d\pi^2(t,\delta)}{dt^2} = -mk(\delta)[2B'(t) - B''(t)t] - FB''(t)C(\delta) \;< 0$$

$$\frac{d\pi^2(t,\delta)}{d\delta^2} = -mk''(\delta)B(t)t - FB(t)C''(\delta) \;< 0$$

$$\frac{d\pi^2(t,\delta)}{dtd\delta} = -mk'(\delta)B'(t)t - mk'(\delta)B(t) - FB'(t)C'(\delta) = -mk'(\delta)B(t)$$

To show that profit function is concave at any optimal *t* and δ, we need to show that

$$\mathrm{H}(t,\delta) = \begin{vmatrix} \dfrac{d\pi^2(t,\delta)}{dt^2} & \dfrac{d\pi^2(t,\delta)}{dtd\delta} \\ & \\ \dfrac{d\pi^2(t,\delta)}{dtd\delta} & \dfrac{d\pi^2(t,\delta)}{d\delta^2} \end{vmatrix} > 0$$

The assumptions required to ensure H(t, δ) > 0 do not have direct economic interpretation. The following example shows, however, that the second condition holds for some simple functional forms. Suppose $B(t) = t^2, K(\delta) = (1-\delta), C(t,\delta) = Ft^2\delta^2$. Then the profit function is $\Pi(t,\delta) = m[V-(1-\delta)t^2]t - Ft^2\delta^2$. Taking the first derivative

$$\pi_t = m[V-(1-\delta)t^2] - 2m(1-\delta)t^2 - 2Ft\delta^2 = 0 \text{ and } \pi_\delta = m\,t^3 - 2Ft^2\delta = 0$$

Similarly, the second derivatives are

$$\pi_{tt} = -6m(1-\delta)t \qquad \text{and} \qquad \pi_{\delta\delta} = -2Ft^2$$

Using the property $m\,t^3 - 2Ft^2\delta = 0$ and simplifying leads to $\pi_{t\delta} = 4Ft\delta$. Therefore the Hessian is

$$H(t,\delta) = \begin{vmatrix} -6m(1-\delta)t & 4Ft\delta \\ 4Ft\delta & -2Ft^2 \end{vmatrix} = 12Fmt^3(1-\delta) - 16F^2t^2\delta^2.$$

Again using the property $m\,t^3 - 2Ft^2\delta = 0$, H(t, δ) = $24F^2t^2\delta(1-\delta) - 16F^2t^2\delta^2$, so that H(t, δ) ≥ 0 if and only if $10\delta \le 6$. For F large enough relative to m, we can ensure that the optimal δ will always satisfy this condition.

Proof of Proposition 1

Consider the objective function

$$\Psi(t,\delta;m) = m[V - k(\delta)B(t)]\,t - FB(t)C(\delta)$$

The first order conditions for *t* and *δ* are

$$\Psi_t = m[V - k(\delta)B'(t)\,t - k(\delta)B(t)] - FB'(t)C(\delta) = 0$$

$$\Psi_\delta = -\,mk'(\delta)B(t)\,t - FB(t)C'(\delta) = 0 \quad \text{Or,} \quad -mk'(\delta)t = FC'(\delta)$$

Taking the total derivative of both equations

$$[-mk(\delta)(2B'(t) - B''(t)t) - FB''(t)C(\delta)]dt + [-mk'(\delta)B(t)]d\delta = [-FB'(t)C(\delta)]dm$$

$$[-mk'(\delta)B(t)]dt + [-mk''(\delta)B(t)t - FB(t)C''(\delta)]d\delta = [k'(\delta)B(t)t]dm.$$

By Crammer Rule'

$$\frac{dt}{dm} = \frac{\begin{vmatrix} -FB'(t)C(\delta) & -mk'(\delta)B(t) \\ k'(\delta)B(t)t & -[mk''(\delta)B(t)t + FB(t)C''(\delta)] \end{vmatrix}}{H(t,\delta)} .$$

It is immediately clear that the numerator is positive because C''(δ) > 0 and k''(δ) > 0. The denominator is positive by assumption. Therefore, $\frac{dt}{dm}$ is positive. In other words, firms will release their products earlier in a large market. Similarly,

$$\frac{d\delta}{dm} = \frac{\begin{vmatrix} [-mk(\delta)(2B'(t) - B''(t)t) - FB''(t)C(\delta)] & -FB'(t)C(\delta) \\ [-mk'(\delta)B(t)] & k'(\delta)B(t)t \end{vmatrix}}{H(t,\delta)}$$ which is again clearly positive because k'(δ) > 0. Therefore, the firms invest more in patching when they enter a larger market.

Proof of Lemma 2

The second order conditions can be written as follows

$$\frac{dL^2}{dt^2} = A1 = \lambda k(\delta)[2B'(t) + B''(t)t] - FB''(t)C(\delta) < 0$$

$$\frac{\partial^2 L}{\partial t \partial \delta} = A2 = \lambda k'(\delta)B(t) > 0 \ ; \qquad \frac{\partial^2 L}{\partial t \partial \lambda} = A3 = \frac{FB'(t)C(\delta)}{\lambda} < 0$$

$$\frac{\partial^2 L}{\partial \delta^2} = A4 = -FB(t)C''(\delta) + \lambda k''(\delta)B(t)t < 0$$

$$\frac{\partial^2 L}{\partial \delta \partial \lambda} = A5 = k'(\delta)B(t)t < 0 \ ; \qquad \frac{\partial^2 L}{\partial \lambda^2} = 0$$

To check the sign of $\frac{\partial \delta}{\partial \overline{q}}$ and $\frac{\partial t}{\partial \overline{q}}$ we use the implicit function theorem and apply Crammer's rule -

$$\frac{\partial t}{\partial \overline{q}} = \frac{\begin{vmatrix} 0 & A2 & A3 \\ 0 & A4 & A5 \\ -1 & A5 & 0 \end{vmatrix}}{D}$$

The numerator is simply –A2A5 + A3A4 which is clearly positive. The denominator is simply

$$D = \begin{vmatrix} A1 & A2 & A3 \\ A2 & A4 & A5 \\ A3 & A5 & 0 \end{vmatrix}$$

which is –A1A5A5 + A2A3A5 + A3 (A2A5 – A3A4) > 0. Therefore, $\frac{\partial t}{\partial \overline{q}}$ is positive.

Similarly,

$$\frac{\partial \delta}{\partial \overline{q}} = \frac{\begin{vmatrix} A1 & 0 & \mathrm{A3} \\ A2 & 0 & \mathrm{A5} \\ A3 & -1 & 0 \end{vmatrix}}{D}$$

The numerator is A1A5-A3A3A5. This is again clearly positive.

QED

References

Abdel-Hamid, Tarek K. (1988), "The Economics of Software Quality Assurance: A Simulation-Based Case Study", *MIS Quarterly*, 12(3): 395-411.

Arthur L. J. (1988), "Software Evolution", *John Wiley & Sons Inc*, New York.

Banker R., G. Davis, and S. Slaughter (1998), "Software Development Practices, Software Complexities, and Software Maintenance", *Management Science*, 44:4, 433-450.

Baron E. (2000), "Mission-Critical Patching: Don't Say Latest and Greatest Patches", *InterWorks 2000 HP Technical Conference.*

Buckley, F.J. (1984), "Software Quality Assurance", *IEEE Transactions on Software Engineering*, 10(1): 36-41.

Brooks F. R. (1995), *The Mythical Man Month*, 2nd Ed. Addison Wesley

Business Week (1999), "Software Hell", December 6.

Capability Maturity Model for Software, Version 1.1, (Paulk, M. C., B. Curtis, M. B. Chrissis and C. V Weber), CMU/SEI-93-TR-24, ADA 263403, *Software Engineering Institute, Carnegie Mellon University,* 1993.

Cohen M.A., Eliashberg, J., and Ho Teck-Hua (1996), "New Product Development: The Performance and Time-to-Market Tradeoff", *Management Science*, 42(2), 173-186.

Dyson E. (1991), “Microsoft’s Spreadsheet, on Its Third Try, Excel”, *Forbes*, 147(7), April 1: 118.

Evans D.S. and B. Reddy (2001), “Government Preferences for Promoting Open-Source Software: A Solution in Search of a Problem”, *working paper*, downloaded from http://ssrn.com on October 2002.

Economic Times (2002), “Microsoft Security Push Cost” $100 Million, July 19.

ISO 9001 Interpreted for Software Organizations, (Radice, Ronald), *Paradoxicon Publishing*, 1995.

Krishnan M S., C. H. Kriebel, S. Kekre, and T. Mukhopadhyay (2000), “An Empirical Analysis of Productivity and Quality in Software Products”, *Management Science,* 46(6): 745-59.

Milgrom P. and J. Roberts (1990), “The Economics of Modern Manufacturing: Technology, Strategy, and Organization”, *The American Economic Review,* 80(3): 511-28.

Musa M. and S. Rosen (1978), “Monopoly and Product Quality”, *Journal of Economic Theory*, 18: 301-317.

NIST Report (2002), “The Economic Impacts of Inadequate Infrastructure for Software Testing”, *National Institute of Standards and Technology*, May.

Ronnen Uri (1991), “Minimum Quality Standards, Fixed Costs, and Competition”, *Rand Journal of Economics*, 22(4): 490-504.

Shaked A. and J. Sutton (1983), “Natural Oligopolies”, *Econometrica*, 51(5): 1469-84.

Viega, J., and G. McGraw (2001), *Building Right Software: How to Avoid Security Problems the Right Way*, Addison Wesley.

Wall Street Journal (1991), “IBM Bends Its Rules to Make a Laptop: New Process Slashes Development Time”, April 6.

Zirger B. and M. Maidique (1990), “A Model of New Product Development: An Empirical Test”, *Management Science*, 36(7), 867-883.