The Influence of Software Process Maturity and Customer Error Reporting on Software Release and Pricing

Terrence August∗
Rady School of Management
University of California, San Diego

Marius Florin Niculescu†
College of Management
Georgia Institute of Technology

Working draft: October 31, 2011
Please do not cite or distribute without permission from the authors.

Abstract
Software producers are making greater use of customer error reporting toward the end of software development processes and throughout the maintenance lifetime of their products. We study how software development differences among producers, such as varying process maturity, and software class and functionality differences, such as operating system versus productivity software, affect how these producers coordinate software release timing and pricing to optimally harness error reporting contributions from users. In settings where prices are fixed, we characterize informative bounds on a software’s optimal release time. Furthermore, contrary to conventional wisdom, we further demonstrate that under certain market conditions it can be preferable to delay release when either customer error reporting rates increase, software functionality decreases, or the size of a pre-release beta testing group increases. When a software producer has limited control over price, we demonstrate that higher process maturity or intermediate maturity, low user error detection contribution, and low functionality tend to provide incentives for the producer to adopt a strategy where it releases its product to the market at the earliest time possible and relies solely on price to shape adoption. However, lower maturity firms may have incentives to delay release when their products exhibit higher levels of functionality. Moreover, we demonstrate that an increase in functionality may be associated with a price markdown for both higher and lower quality products. On the other hand, in settings where the producer has full price control, regardless of functionality, it will always release the software as early as possible and rely solely on pricing to control adoption. Further, we establish that higher process maturity producers targeting markets with low error reporting rates should employ price skimming strategies toward the end of their product sales horizons and that penetration pricing strategies tend to be optimal early on in product life cycles. Such penetration strategies are also used by lower maturity producers that develop software in classes characterized by high user error reporting rates.

∗Rady School of Management, University of California, San Diego, La Jolla, CA 92093-0553. e-mail: taugust@ucsd.edu
†College of Management, Georgia Institute of Technology, Atlanta, GA 30308, e-mail: marius.niculescu@mgt.gatech.edu
1 Introduction

Over the last decade, the world has achieved phenomenal growth in broadband penetration. For example, the United States rose from 4.4% penetration in 2001 to 26.7% by 2009 – a sixfold increase (OECD 2009). The Organization for Economic Co-operation and Development also reports that, averaging across the G7 countries, the penetration was approximately eighteen times higher by the end of the decade.\(^1\) This extensive growth in high speed access internationally has presented both newfound opportunities and challenges. With now over 600 million interconnected Internet hosts and over 2 billion Internet users across the world, the ease with which businesses, individuals, and society at large can communicate and exchange information is unprecedented (ISC 2010, Internet World Stats 2011). Notwithstanding these benefits, the interconnectedness of computers and the speed at which information disseminates across the Internet are actively exploited by malicious individuals to enhance the spread of worms, viruses, and other software designed to cause substantial economic damages. For example, banks and consumers are regularly victims of fraud and identify theft as a result of their electronic transactions (Pilon 2010, Richmond 2010). Similarly, software products critical to the creation and exchange of information such as Microsoft Windows, McAfee AntiVirus, and Adobe Reader are heavily utilized by users yet suffer from a constant stream of flaws and vulnerabilities (Claburn 2008, McMillan 2009, McCullagh 2010). According to RTI (2002), the annual cost of faulty software to the U.S. economy alone was $59.5 billion roughly a decade ago and has since grown to an estimated $75 billion (Michaels 2008). Hence, an ongoing challenge faced by both private entities and the public sector is how to continue to obtain increased value from this global interconnected network while mitigating the associated negative aspects.

Recently, software firms have begun to leverage consumers’ connectedness to the Internet to enable their users to contribute toward increasing the reliability and security of their software products. Applications now routinely have built-in support for automatic error reporting through which application failure information is transmitted from users’ systems to software firms. Microsoft’s Windows Error Reporting, AutoCAD’s Customer Error Reporting, Mozilla’s Crash Reporter, and other similar implementations all aim to harness the diverse user population’s idiosyncratic environments and behavior to help with software assurance (Markoff 2006). Through these tools, software firms collect substantial input from the user community regarding product failures in the field of operation. For example, Mozilla gets 2.5 million crash reports per day (Thompson

\(^1\)Globally, countries are continuing to make substantial investments to further promote this growth and foster economic development. As part of the American Reinvestment and Recovery Act in 2009, the U.S. appropriated $7.2 billion toward expanding broadband access in underserved areas, and the United Kingdom has also made a commitment to delivering broadband into every household by 2012 (Tryhorn 2009, Ransom 2010). Countries leading the way such as South Korea already have much higher penetration and can also offer significantly faster broadband speeds (Sutter 2010).
In many instances, software testers cannot reproduce customer discovered errors using identical input because factors involving “invisible users” or the interactions between the software and the customer’s environment (operating system, file system, and libraries) are the actual root cause (Whittaker 2001). Hence, error reports generated by users’ systems, which often include a snapshot of important environment information, have a significant potential to reduce the costs of correcting bugs; these costs can account for up to 50-75% of software development costs (Muthitacharoen and Saeed 2009). In a memo to customers in October 2002, Steve Ballmer, CEO of Microsoft, notes that “in Windows XP Service Pack 1, error reporting enabled [Microsoft] to address 29 percent of errors involving the operating system and applications running on it, including a large number of third-party applications. Error reporting helped [Microsoft] to eliminate more than half of all Office XP errors with Office XP Service Pack 2” (Ballmer 2002).

With modern software assurance strategies that leverage users’ systems and their interconnectedness, a software firm must carefully select product release dates that account for several important trade-offs. On one hand, by releasing earlier, the firm’s product is available for a longer time in the market before it becomes outmoded; the firm reduces its cost of detecting and fixing software bugs due to error reporting; and the firm may even enjoy competitive benefits associated with being first to market (Cohen et al. 1996). On the other hand, earlier released software bears a lower initial quality which has several effects: users incur costs associated with security attacks and poor application performance in the interim; users, in turn, impose goodwill costs on the firm; and the speed at which consumers adopt the software, i.e., the rate of diffusion, is reduced (Keizer 2007).

These aforementioned trade-offs will be strongly influenced by the distinct characteristics of the firm, its product, and the corresponding market. One particularly relevant characteristic of a software firm is its software process maturity which is “...the extent to which a specific process is explicitly defined, managed, measured, controlled, and effective” (Paulk et al. 1993). The federally funded Software Engineering Institute (SEI) established a set of Capability Maturity Models (CMMs) to help organizations improve processes with emphasis on those related to software development. Some of these basic models were superseded by SEI’s Capability Maturity Model Integration (CMMI), but the essence is much the same. CMMI defines five levels of maturity in software-producing organizations that range from Level 1, which is characterized by ad hoc processes that can complete goals, to Level 5, where processes are already rigorously defined and managed in a quantitative way, and the focus now lies on optimizing them (SEI 2006). One useful aspect of the CMMI is that firms at the same level tend to be similar on other dimensions as well. For example, Table 1 illustrates the relationship between a firm’s software process maturity and the quality of its software, measured in terms of defects per thousand lines of code. This positive
Table 1: The relationship between the average number of defects per thousand lines of code and an organization’s level achievement in the Capability Maturity Model (Davis and Mullaney 2003).

<table>
<thead>
<tr>
<th></th>
<th>CMM Level 1</th>
<th>CMM Level 2</th>
<th>CMM Level 3</th>
<th>CMM Level 4</th>
<th>CMM Level 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Defects/KLOC</td>
<td>7.5</td>
<td>6.24</td>
<td>4.73</td>
<td>2.28</td>
<td>1.05</td>
</tr>
</tbody>
</table>

relationship between process maturity and quality as well as other software development characteristics has also been established in the academic literature (see e.g., Harter et al. 2000, Harter and Slaughter 2003). Hence, by understanding how bug density interacts with the trade-offs that affect software release which are described above, we can provide broad implications on how firms at a given level of process maturity should manage software adoption and even identify the value that underlies achieving a higher level of maturity.

A second important factor that affects how a software firm manages adoption through its release timing and price is the amount of software functionality built into a product. One observation is that, over time, an increasing amount of functionality is being included with each new version of a given software product. For example, Microsoft Windows versions, NT 4.0, 2000 Professional, XP Professional, and Vista, have 16, 29, 40, and 50 million lines of code, respectively (Henry 2007, Manes 2007). Additionally, different classes of software can be correlated with different amounts of functionality. For example, for another mainstream operating system software, Wheeler (2001) studies Red Hat Linux 7.1 which was released around the same time as Windows XP and had a comparable 30+ million lines of code. Popular open source database software products PostgreSQL and MySQL are estimated to have approximately 900 thousand lines of code each (Babcock 2008, MST 2010).

In this paper, using an analytical model, we formally examine how a software firm adjusts its release timing and pricing in a setting where it can harness customer error reporting. In this context, we specifically study the influence of the firm’s pricing power, its software process maturity, and the characteristics of the product class where it intends to compete as well as its relative quality within that class on its timing and pricing decisions. In the next section, we describe how our model fits with the rest of the literature while highlighting our focus on the intimate connection between how the firm adjusts the evolution of adoption and the strength and need of error reporting contributions. In Section 3, we formally present the model and then explore both exogenous price and optimal static price settings in Section 4. In section 5, we study an extension where the firm optimizes a price trajectory, and Section 6 offers our concluding remarks.

---

2The two studies use different metrics; for MySQL, a measure of the effective lines of source code was utilized.
2 Literature Review

This study is directly related to the literature on optimal software release time. A vast portion of the literature studies the firm’s strategy from a maintenance cost minimization perspective, independent of software demand. Examples include Okumoto and Goel (1980), Koch and Kubat (1983), Yamada et al. (1984), Yamada and Osaki (1986, 1987), Hou et al. (1997), Kimura et al. (1999), Pham and Zhang (1999), Zheng (2002), Xie and Yang (2003), and Huang and Lyu (2005). Several studies (see, e.g., Dalal and Mallows 1988, McDaid and Wilson 2001, Ji et al. 2005, Jiang et al. 2011) extend this framework and incorporate opportunity costs of lost sales due to late release through a price-invariant cost component that increases in the time to market. Recently, Arora et al. (2006) implement a total sales function that depends on release time and patching levels, and, in the current study, we further permit time-varying pricing and the presence of discounting and network effects. One common characteristic of extant models in the literature is that the expected evolution of software adoption does not impact the firm’s software release decision, and we relax this assumption in our work. While some more general studies on innovation release (e.g., Kalish and Lilien 1986) do account for the shape of the adoption curve, we further advance our understanding of this topic by focusing on the software industry and its idiosyncrasies including quality improvement dissemination via patching, negligible reproduction costs, and, importantly, customer error reporting.

Aiming to clarify and study the tradeoff between internal testing and debugging costs on one hand, and revenues, goodwill penalties, and consumer error reporting benefits on the other, we complement the existing literature on optimal software release modeling by employing a continuous-time parameterization of software demand and incorporating quality and network effects on adoption. The shape of the adoption curve impacts a firm’s profit in multiple ways. First, under time discounting, it directly affects revenues. Second, it impacts consumers’ aggregate contribution to bug detection and the firm’s post-release testing efforts. Many past studies on software release examine a setting where the firm ceases to test the software for flaws after bringing it to the market and implicitly incorporate consumers’ participation to error reporting by assuming a bug fixing cost associated with flaws discovered in the field of operation, which, in the absence of in-house detection efforts, are reported only by users. To the best of our knowledge, Jiang et al. (2011) propose the first model that explicitly incorporates consumer error reporting and considers the continuation of in-house testing beyond release time. In their model, the increase in the bug detection rate for any remaining bug due to consumers using the software is assumed constant throughout the product life cycle, and the testing cost is linear in the testing time. We extend their work by further incorporating the effect of software demand on both dimensions. Specifically, we allow the consumers’
contribution to the bug detection rate to grow with the installed base and parameterize the actual error reporting rate to account for users opting out of crash reporting due to privacy concerns.

Third, there is an explicit link between the cost to firms of addressing post-release software bugs and how the network size evolves over time, which is another important contribution of our work. Prior models typically consider these costs as being linear in the number of bugs that are detected after market introduction. Some studies account for it more abstractly as a function of time (Shantikumar and Tufekci 1983), reliability (Pham and Zhang 1999), or the number of remaining flaws (Ji et al. 2005). Ehrlich et al. (1993) also introduce a cost to the software firm resulting from consumer use that depends on the software failure intensity at release, the usage period, and an exogenous demand that is independent of the model parameters. In the current work, beyond post-release bug fixing costs, we further incorporate that the firm also incurs goodwill costs at a rate that is proportional to both the current bug count and network size. In this manner, we are able to capture the fact that an error that is detected early after release, when there are few adopters, is likely to be less costly to the firm compared to an error that resides in the code longer and generates greater damage to a larger consumer base. Over time, the network size increases and the number of resident errors decreases, generating an important dynamic with the goodwill cost rate and how the firm controls adoption.

Building on the above modeling contributions, in this work, we advance theory on how software firms strategically manage adoption through release time and pricing. We investigate settings in which the firm has incentives to release earlier or later and how it accelerates or decelerates adoption through pricing as influenced by consumer error reporting, the firm’s software process maturity, software functionality, and adoption-dependent costs. For example, under limited price control, optimal release time and pricing can be either increasing or decreasing in the consumer error reporting rate or functionality level depending on relationships between testing, processing, fixing, and goodwill costs. In the remaining sections, we study the interaction between these critical elements of software production and adoption to shed light on how software should be brought to market in a variety of settings.

3 The General Model

A firm offers a software product licensed for perpetual use and supports it until its product discontinuation time $T > 0$. Beyond $T$, consumers who have already purchased may continue to use the

---

3 In the rapidly evolving software industry, this discontinuation time is often exogenously determined by the rate of technological advance. In other cases, due to future major releases of a product that are developed in parallel, the discontinuation time can be the result of planned functional obsolescence. For simplicity, in our model, we take $T$ as fixed and focus attention on how community contributions (bug detection) to improve quality affect a software
software which still has value, but the firm ceases all quality improvement efforts. The software product’s extent of features and overall complexity is related to its software class, ranging from enterprise software to simple end-user applications. We denote the software product’s complexity as $Y > 0$ which can be interpreted as the the size of the product’s codebase or its number of basic units of code. Stemming from the CMMI and implementation in industry, a standard measure of software quality is the inverse of defect density, i.e., defects per thousand lines of code (Lockheed Martin 2003, Northrop Grumman IT 2003). We assume that the software product has $\bar{B}$ bugs or defects at the earliest moment it can feasibly be released, denoted $t = 0$, which is often considered to be the beginning of the release candidate stage where features are no longer being added and the focus turns to final testing and debugging (Petreley 1998). Therefore, using this standard measure, initial software quality is given by $Y/\bar{B}$.

We denote the software process maturity of the firm with $\gamma \in [0, 1]$. Recent empirical studies (Harter et al. 2000, Krishnan et al. 2000, Harter and Slaughter 2003) document a positive relationship between process maturity and software quality. Following both industry evidence and the convention in the literature, we also assume this relationship, $\bar{B}/Y = \sigma(\gamma)$, exists where we use defect density for convenience and $\sigma(\gamma)$ has the following properties: $\partial \sigma / \partial \gamma < 0$, $\sigma(0) = \bar{\sigma}$, and $\sigma(1) = 0$. Taken together, firms with more mature software development processes (higher $\gamma$) arrive at the release candidate stage with lower bug density (i.e., $\bar{B}/Y$ is decreasing in $\gamma$). Both the product’s life cycle $0 < T < \infty$ and the market potential $0 < m(Y) \leq \bar{m} < \infty$ are finite, the latter being weakly increasing in the complexity or level of functionality provided by the software.

Let $D(t)$ denote the number of unique bugs detected and reported by time $t \geq 0$, with initial condition $D(0) = 0$. We assume a bug detection process following properties of the mean of the classic non-homogeneous Poisson process model for software reliability in Goel and Okumoto (1979), whereby at any given time $t$, the rate at which previously undiscovered bugs are detected is proportional to the number remaining in the code. This model and variants of it have been widely used in the research on optimal software release timing (see, e.g., Dalal and Mallows 1988, Ehrlich et al. 1993, Xie and Hong 1998, Pham and Zhang 1999, Jiang et al. 2011). More precisely, we consider:

$$\frac{\partial D(t)}{\partial t} = \theta(t) \times (\bar{B} - D(t)),$$

(1)

producer’s release timing and pricing.

In particular, consumers no longer impose goodwill costs on the firm after $T$. Realistically, consumers could still impose some goodwill penalties on the firm after this point in time, but these costs would be limited in comparison to those imposed as a result of poor quality due to software defects during the active life of the product.
where $\theta(t)$ can be interpreted as the overall rate of detection per undiscovered bug. Both the firm and existing users can contribute to the bug detection process. The firm contributes $\theta_f(t) \geq 0$ to the overall rate by incurring testing effort, and any adopter who chooses to provide quality feedback contributes $\theta_u(Y) \in [0, \bar{\theta}_u]$. We assume consumers use the software uniformly over time and across functions, but that their usage varies with the software’s level of functionality $Y$ which, in turn, affects the detection rate. Because consumer error reporting is usually optional, with some users electing not to participate due to concerns over privacy and possible work disruption (Muthitacharoen and Saeed 2009), we denote the portion of users who participate with $\alpha \in [0, 1]$. Thus, if $N(t)$ denotes the number of existing users at time $t$, then the overall detection rate per undiscovered bug is given by:

$$\theta(t) = \theta_f(t) + \alpha N(t)\theta_u(Y).$$  \hspace{1cm} (2)

We denote the number of bugs still resident (either undetected or detected but unfixed) in the code as $B(t)$ with initial condition $B(0) = \bar{B}$. Once a bug is reported, it is assigned to a pool of bugs that have been detected but not fixed yet. At time $t$, this pool contains $D(t) - (\bar{B} - B(t))$ reported defects, and the firm works toward addressing them by issuing patches at a rate $\xi$ per unfixed defect. Hence, the rate of change for remaining bugs in the code is given by:

$$\frac{\partial B(t)}{\partial t} = -\xi \times (D(t) - \bar{B} + B(t)).$$ \hspace{1cm} (3)

We implicitly capture the property that bugs are likely to be heterogeneous in the amount of effort required for each to be resolved. In particular by the dynamics in (3), once detected, some bugs will be patched fast, others will take longer to be fixed, and some known flaws will not be removed from the code prior to product discontinuation.

Drawing on the vast literature on innovation diffusion sparked by Bass’s model (1969), we parameterize the evolution of the cumulative installed base of users $N(t)$ through a continuous time hazard rate model as follows:

$$\frac{\partial N}{\partial t}(t) = (m(Y) - N(t)) \left( a + b \frac{N(t)}{m(Y)} - c \frac{B(t)}{Y} \right) w(p(t)), \hspace{1cm} (4)$$

over $t \in [t_0, T]$ where $t_0$ is the firm’s choice for release time of the software and $a > 0$. Both positive network effects associated with the software and quality effects determined by its reliability affect the evolution of the cumulative installed base of users. 

---

5This model implicitly captures bug heterogeneity with regard to detection. Starting with $\bar{B}$ bugs initially, some bugs will surface quickly while others will take longer to be detected; some will even go undetected for the entire planning horizon.
the rate of adoption. The relative strength of the network effects is given by $b > 0$. Similarly, we denote the extent to which poor software reliability affects adoption by users with $c > 0$.

Consumer price sensitivity is reflected by a twice differentiable, multiplicative price response function $w(p)$ satisfying (i) $w(p) \geq 0$, (ii) $w'(p) < 0$, (iii) $\lim_{p \to \infty} pw(p) = 0$, and (iv) $\frac{w(p)w''(p)}{(w'(p))^2} < 2$. Our choice of a multiplicatively separable price effect is consistent with the literature on the diffusion of innovation (see, e.g., papers by Robinson and Lakhani 1975, Dolan and Jeuland 1981, Kalish 1983, Thompson and Teng 1984, Bass et al. 1994, Mesak and Clark 1998, Krishnan et al. 1999, Sethi and Bass 2003, Dockner and Fruchter 2004). The third condition implies that, for any functionality, above a certain price point both revenues and adoption are negligible. The last condition simply ensures that the firm’s profit maximization problem has a unique solution.\footnote{Similar technical assumptions are made by Kalish (1983) and Mesak and Clark (1998).}

As an example, negative exponential functions of the form $\tau e^{-\psi p}$ with $\tau, \psi > 0$ satisfy all assumed properties on $w(\cdot)$.

The firm incurs four types of costs: testing, error report processing, bug fixing, and goodwill. Each of these, we discuss in order. First, we assume that the firm broadly tests the entire codebase which is to say it does not know a priori where the defects are located. The firm will incur a total cost rate of $C_T \times \theta_f(t) \times Y$, where $C_T \geq 0$, to induce a bug detection rate (resulting from the firm’s effort) of $\theta_f \cdot (B - D(t))$ through testing of all $Y$ units of the codebase. We assume a discounting rate of $\beta \geq 0$ in the market. Thus, the total testing cost is given by:

$$\tilde{C}_T = \int_0^T C_T \times \theta_f(t) \times Y \times e^{-\beta t} dt.$$  \hspace{1cm} (5)

Second, we assume the firm incurs error processing costs associated with analyzing error reports, assessing the extent of any given defect, and assigning their reparation to an appropriate developer team. User-generated reports usually vary in the quality of information provided and may require additional effort in terms of interpreting a report and reproducing an error (Bettenburg et al. 2008). For these reasons, we denote the per-report processing costs as $C_{P,f} \geq 0$ and $C_{P,u} \geq 0$, depending on whether the report originates from the firm’s testing or users’ reporting, respectively, with $C_{P,f} < C_{P,u}$. Of the rate of newly detected bugs, $\frac{\partial D(t)}{\partial t}$, a fraction $\frac{\theta_f(t)}{\theta(t)}$ originate from the internal team while the remainder $\frac{\alpha N(t)\theta_u(Y)}{\theta(t)}$ are reported by users. Thus, the total processing cost is given by:

$$\tilde{C}_P = \int_0^T \left[ C_{P,f} \times \frac{\theta_f(t)}{\theta(t)} + C_{P,u} \times \frac{\alpha N(t)\theta_u(Y)}{\theta(t)} \right] \times \frac{\partial D(t)}{\partial t} \times e^{-\beta t} dt.$$  \hspace{1cm} (6)

Third, the firm incurs quality improvement costs by allocating effort to resolve bugs that have
been reported but are still unfixed at a cost rate $C_F \geq 0$. For simplicity, we assume the firm works concurrently on fixing all $D(t) - \bar{B} + B(t)$ defects. By (3), some fixes are issued quickly while others become delayed. As a result, the firm can receive duplicate error reports which must be cross-checked with the bug tracking system at a cost $C_D \geq 0$ before being discarded. While pending resolution, any detected but unresolved bug can still be re-detected at rate $\theta(t)$, resulting in a duplicate report cost rate of $C_D \times \theta(t) \times (D(t) - \bar{B} + B(t))$. Therefore, the total bug fixing cost is given by:

$$\tilde{C}_F = \int_0^T (C_F + \theta(t)C_D) \times (D(t) - \bar{B} + B(t)) \times e^{-\beta t} dt.$$  \hspace{1cm} (7)

Finally, the firm incurs goodwill costs from exposing its users to buggy software. Software crashes typically generate damage to users only when they occur. However, not all flaws lead to crashes or system alerts. For example, security breaches can go undetected without any system interruption, and malicious hackers can exploit security vulnerabilities repeatedly before they are detected and patched. We denote the average cost incurred per each user, bug, and unit of time with $C_G \geq 0$, and the total expected goodwill costs are given by:

$$\tilde{C}_G = \int_{t_0}^T C_G \times N(t) \times B(t) \times e^{-\beta t} dt.$$ \hspace{1cm} (8)

Because software is a digital good, we make the traditional assumption that there are no capacity constraints and the marginal cost of reproduction is zero. Revenues are generated at each point in time when a user adopts the product, hence the firm’s expected discounted profit can be written

$$\Pi(t_0, p(\cdot), \theta_f(\cdot)) = \int_{t_0}^T p(t) \times \frac{\partial N(t)}{\partial t} \times e^{-\beta t} dt - (\tilde{C}_T + \tilde{C}_P + \tilde{C}_F + \tilde{C}_G).$$ \hspace{1cm} (9)

Using the framework laid out in this section, we can begin to explore how firms of varying process maturity and with different classes of software products should manage adoption through release timing and pricing.

4 Managing Adoption and Error Reporting

Our primary goal in this paper is to better understand how software firms, which have diverse products and varying levels of process maturity, can utilize customer error reporting as part of their overall strategy to simultaneously manage quality improvement costs and adoption. Going forward, we will use a blend of analytical results and numerical analysis to generate insights from the complex
interactions underlying the model presented in Section 3. In the most general form, in addition to release time and price, the firm’s detection rate \( \theta_f(t) \) can be optimized as a control variable for the firm. However, for the model to remain tractable, we will make a simplifying assumption that the overall detection rate \( \theta(t) \) is constant which is to say that the firm’s detection rate decreases as error reporting contributions increase due to adoption in an exogenous manner. This assumption facilitates our ability to generate important analytical insights connected to our research questions, which will prove to be quite robust. In Section 4.1.3, we will relax this assumption and perform extensive numerical studies where \( \theta_f(t) \) is optimized for comparison and discussion.

For now, suppose \( \theta \) is constant and satisfies \( \theta \geq \alpha m(Y) \theta_u(Y) \) for any feasible functionality level. Then, by (1), the number of unique bugs that have been detected by time \( t \), whether fixed yet or not, is given by

\[
D(t) = \bar{B} \times \left( 1 - e^{-\theta t} \right).
\]

Using (3) and (10), the number of bugs still remaining in the software, either undetected or detected but not yet fixed, exhibits the following the trajectory over time

\[
B(t) = \bar{B} \times \Gamma(t),
\]

where \( \Gamma(t) \triangleq \frac{1}{e^\theta - \theta} (\xi e^{\theta t} - \theta e^{-\xi t}) \). By (4), a necessary condition for adoption to begin is that the hazard rate must be positive, i.e. \( a \geq \frac{cB(t_0)}{Y} \). Using (11) and the relationship between software maturity and initial bug density, this condition is equivalent to \( a \geq c \sigma(\gamma) \Gamma(t_0) \). Because \( \Gamma(t) \) is decreasing and \( \Gamma(0) = 1 \), adoption can start immediately at \( t = 0 \) if \( a \geq c \sigma(\gamma) \). Otherwise, because \( \lim_{t \to \infty} \Gamma(t) = 0 \), there exists a unique bound \( \tilde{t} \) satisfying \( a = c \sigma(\gamma) \Gamma(\tilde{t}) \) such that adoption can start provided \( t_0 \geq \tilde{t} \). Summarizing, the release time must satisfy

\[
t_0 \geq L(\gamma) \triangleq \begin{cases} 0, & \text{if } a \geq c \sigma(\gamma); \\ \tilde{t}, & \text{otherwise}. \end{cases}
\]

We refer to (12) as the release time constraint. It essentially reflects observed software industry behavior where release candidate stages or late beta stages can last extended periods of time, illustrating the difference between what the firm might initially consider a releasable product and the actual needs and perceptions of the consumers. Since there is a bijective mapping between release time and expected bug count at release time, this constraint can be seen as similar to the software reliability/quality constraint at release time imposed in other papers such as Yamada and Osaki (1987), Xie and Hong (1998), Kimura et al. (1999), and Zheng (2002).
Remark 1 \( L(\cdot) \) is weakly decreasing.

Remark 1 establishes that software firms with higher maturity are able to release their products early to the market and still induce adoption. However, lower maturity firms must delay a bit longer until their product quality is sufficient to induce adoption. In the following, we shall see that it is sometimes in the firm’s best interest to release as soon as it can while still inducing adoption, i.e., \( t_0 = L(\gamma) \); Remark 1 highlights that even this time is firm-specific, reflecting the maturity of internal development processes. Throughout the paper, we focus on the cases when \( L(\gamma) < T \).

4.1 Exogenous Pricing

First, we study the case where price is fixed over the software product’s selling horizon and is determined by the market. Let \( p \) be the price of the software for all \( t \geq t_0 \). Then, it can be shown (derivation included in the Appendix) that the adoption path \( N(t) \) is characterized by a Riccati partial differential equation with solution

\[
N(t) = \begin{cases} 
0 & \text{if } t < t_0; \\
 m(Y) \left(1 + \frac{1}{\nu(t)}\right) & \text{if } t \geq t_0,
\end{cases}
\]

where

\[
\nu(t) = e^{w(p)\rho(t|\cdot)} \times \left( -e^{-w(p)\rho(t_0|\cdot)} + w(p) b \int_{t_0}^{t} e^{-w(p)\rho(x|\cdot)} dx \right)
\]

and

\[
\rho(t|\theta, \xi, \gamma) = (b+a)t + \frac{c\sigma(\gamma)\theta\xi}{\xi - \theta} \left( e^{-\theta t} - e^{-\xi t} \right).
\]

As a result, the adoption path exhibits certain monotonicity properties.

Lemma 1 For all \( t \geq t_0 \),

(i) \( N(t) \) is decreasing in \( t_0 \) and \( p \).

(ii) For a given software product with complexity \( Y \), \( N(t) \) is increasing in \( \gamma \).

(iii) For a given software firm’s process maturity level \( \gamma \), \( N(t) \) is increasing in \( Y \). That is, a firm derives a stronger cumulative adoption from releasing a higher functionality product despite it containing a larger number of bugs.
Part (i) of Lemma 1 is driven by scaled stochastic dominance since the market penetration level $F(t) = N(t)/m(Y)$ behaves like a distribution function ($F(t_0) = 0$, $\lim_{t \to \infty} F(t) = 1$, and $F$ is weakly increasing in $t$ over $[t_0, \infty]$). The pointwise monotonicity of the adoption curve with respect to release time (i.e., cumulative demand $N(t)$ is decreasing in $t_0$) has important implications. Visually, a delayed release adoption path always lies below an early release adoption path. As a consequence, if the firm ever chooses to delay its release in order to increase initial quality and gain stronger initial adoption momentum, this delayed strategy will never make up the lost ground and reach the same cumulative sales volume. In that sense, the quality effect associated with delaying is not stronger than the accumulation of positive network effects associated with an earlier release. Applying integration by parts to the firm’s revenue function, it immediately follows that

$$p \int_{t_0}^{T} \partial N \frac{\partial N}{\partial t}(t)e^{-\beta t} dt = pN(T)e^{-\beta T} + p\beta \int_{t_0}^{T} N(t)e^{-\beta t} dt.$$ 

Hence, due to time discounting and the fact that $N(t)$ is decreasing in $t_0$, revenues are also decreasing in $t_0$, which suggests that any strategic delay of the release by firms to increase initial quality must be driven by cost considerations.

Parts (ii) and (iii) of Lemma 1 provide basic implications on how a firm’s software process maturity or a given software product market (in terms of how much functionality or complexity is typical of software in that market) directly affects adoption. Part (ii) of Lemma 1 states that a software product with a given level of functionality has fewer bugs when it is developed by a firm with more mature software development processes, and, hence, exhibits increased cumulative adoption. This property of the adoption curve suggests that firms with lower process maturity may be more inclined to strategically manipulate release timing to temper these quality effects.

In the last part of the lemma, we highlight that, all else being equal, the net effect of increased software functionality is greater adoption despite the existence of more bugs in the software. Examining equation (4) which governs the diffusion of adoption, it can be seen that an increase in functionality $Y$ increases both the potential market for the software $m(Y)$ and the initial number of bugs $\bar{B} = \sigma(\gamma)Y$. Multiplying the remaining market potential factor times the network effect yields $(m(Y) - N(t)) \times b \frac{N(t)}{m(Y)} = bN(t) \left(1 - \frac{N(t)}{m(Y)}\right)$, thus although the fraction of the market using the software goes down with an increase in $Y$, the net effect of a larger remaining potential is to increase the rate of adoption. Our assumption that bug density slows adoption suggests that a software firm with a given maturity $\gamma$ can increase functionality without hurting adoption provided it maintains this level of capability in its development processes. That is, more bugs certainly get introduced, but additional functionality serves to counterbalance them. Overall, the adoption model we utilize and present in (4) intends to capture characteristics typical of current software markets where producers focus on new features while maintaining reasonable levels of security and reliability; improvements on the latter, although important, is often a secondary concern (Perrin...
4.1.1 Release Time Bounds

Next, we study the profit maximization problem for the firm as it selects the optimal release time. Given a fixed price \( p \), the optimal release time \( t^*_0 \) satisfies

\[
    t^*_0 = \arg\max_{t_0 \in [L(\gamma) , T]} \Pi(t_0).
\]

Although it is not possible to give a complete, explicit closed-form solution to this problem, we can characterize informative bounds on the firm’s optimal release time and subsequently identify conditions under which \( t^*_0 \) is precisely equal to each of these bounds. Outside of these conditions, we show that choosing a release time at a bound can often still generate profits comparable to those obtained under the optimal interior choice for release time. However, first, we present these relevant bounds in the following proposition.

**Proposition 1** The optimal release time for a firm’s software product satisfies \( t^*_0 \in [L(\gamma) , H(Y, \gamma, p)] \), where

\[
    H(Y, \gamma, p) = \begin{cases} 
    L(\gamma), & \text{if } 0 \leq \Phi(L(\gamma)) \\
    \bar{t}, & \text{if } \Phi(L(\gamma)) < 0 < \Phi(T) \\
    T, & \text{otherwise}.
    \end{cases}
\]

Here, \( \Phi(t) \triangleq p\beta + \theta_u(Y)\alpha C_T Y - \theta_u(Y)\alpha(C_{P,u} - C_{P,f})Be^{-\eta t} - C_G B(t) \) and \( \bar{t} \) is the solution to \( \Phi(t) = 0 \), which exists and is unique in the given region. Moreover, if \( H(Y, \gamma, p) < T \), then profits are decreasing in \( t_0 \) over the interval \([H(Y, \gamma, p), T]\).

Proposition 1 conveys several managerial implications. First, even if a software firm cannot precisely determine its optimal release time, it should never delay beyond \( H(Y, \gamma, p) \). It can be shown that for any adoption curve, within the interval \([H(Y, \gamma, p), T]\), late adopters are less valuable to the firm than earlier adopters. Also, in spite of potentially inducing higher sales on some time intervals, a delayed release corresponds to a weaker aggregate installed base pointwise, as seen in Lemma 1. It immediately follows that bringing a software product to market later than \( H(Y, \gamma, p) \) is suboptimal. Proposition 1 also asserts that a firm should release immediately if any constraining factors prevented release prior to \( H(Y, \gamma, p) \) since expected profits are strictly decreasing thereafter. Software firms sometimes face such constraints when the adoption of their product is tied to the availability of specific hardware. For example, in the spring of 2006, Sony officially announced a
delay in the release of its Play Station 3 console until November due to unresolved issues with the production of Blu-ray components (Nagai 2006). In such a case, some of the firms who would have originally delayed release of their titles to improve initial quality may no longer have any remaining incentives to further delay under the new console launch date.

Firms are already aware that the release time must occur in interval $[L(\gamma), T]$. Thus, the derived upper bound $H$ increases in managerial relevance as the ratio $\frac{H-L}{L}$ becomes smaller. Depending on parameters, this ratio can take on any value in the interval $[0, 1]$. In particular, when $H = L$, our derived bounds precisely identify the optimal release time. Additionally, as a firm increases the maturity of its development processes, both bounds on the optimal release time have a tendency to shrink (see Remark 1, and, similarly, $\partial H/\partial \gamma \leq 0$ follows directly from (17)). This suggests that an earlier release may be preferable. Another distinct benefit of the bounds characterized in Proposition 1 is that they help expedite numerical computation of exact solutions to (16) by significantly reducing the search space. Beyond the bounds being useful and informative, we next further characterize the optimal release time, establishing conditions where it either equals or is sufficiently close to one of the bounds.

**Corollary 1** When error reporting contributions from users are large substantial and a firm possesses high software maturity, i.e., $\Phi(L) \geq 0$ is satisfied, then $t^*_0 = L = H$.

Corollary 1 follows directly from Proposition 1, stating that the bounds effectively sandwich $t^*_0$ to satisfy $L$ when the condition $\theta_u(Y)\alpha \left( C_T Y - (C_{P,u} - C_{P,f})\sigma(\gamma)Ye^{-\theta L(\gamma)} \right) \geq C_G\sigma(\gamma)Y \times \Gamma(L(\gamma)) - p\beta$ is satisfied. As firm maturity $\gamma$ increases, $\sigma(\gamma)$ decreases which dampens $(C_{P,u} - C_{P,f})\sigma(\gamma)Ye^{-\theta L(\gamma)}$ and, together with sufficient error reporting contributions from users $\theta_u(Y)\alpha$, yields the result. In this case, the firm’s benefit from alleviating internal testing costs outweigh the additional costs of processing user-reported bugs, which are limited further limited by the firm’s high development maturity. In Figure 1, we illustrate how the optimal release time compares to the derived bounds while deviating a single parameter from a base case in each panel on the left. Panels (a), (c), and (e) all demonstrate that for sufficiently large $\gamma$, the bounds have collapsed suggesting that the firm should release its product at the earliest possible point in time that can generate adoption. Said differently, firms that utilize mature development processes (e.g., those who have obtained CMM Level 5) should release their products to the market as soon as they have a release candidate available. Although there are always some negative effects on adoption stemming from reduced quality due to bugs, at their level of maturity, releasing immediately to boost adoption through network effects is a preferred strategy.
Proposition 2 When error reporting contributions from users are small and a firm’s development processes are not highly mature, i.e., $\Phi(L) < 0$ is satisfied, then $L < H$. Furthermore, in this case:

(i) If maturity is very low (i.e., $\sigma(\gamma) > z_1 \triangleq \frac{a-b}{c}$), then there exists $\bar{T} > 0$ such that $t_0^* = L$ when $T < \bar{T}$.

(ii) For any $\epsilon > 0$, if the product exhibits low network effects, the firm’s process maturity is not too low ($\sigma(\gamma) < z_1$), $p$ and $C_T$ are low enough, and $T$ is sufficiently high, then $\frac{H-t_0^*}{T-L} < \epsilon$.

Part (i) of Proposition 2 states that if a firm’s maturity is low, then it should also release products with shorter lifecycles at $L(\gamma)$, but the reasoning is different in this case. As established in Remark 1, firms with lower maturity necessarily must release their software products later before
adoption can occur. Thus, it is not the case that a lower maturity firm should release software products early in the absolute sense. Rather, it should release its product as soon as it feasibly can because by the time it has a release candidate with quality sufficiently high to induce adoption, there is little time remaining before product discontinuation. Panel (e) of Figure 1 illustrates how $L(\gamma)$ decreases in software maturity. As $\gamma$ becomes small, the remaining sales horizon is extremely limited and the firm should optimally release at $L$. As $\gamma$ increases, $T - L(\gamma)$ increases and the firm’s sales horizon is less constrained. In this case, it may prefer to delay release in order to improve quality first which is depicted by $t^*_0$ moving into the interior region between bounds $L$ and $H$. However, as $\gamma$ continues to increase, the firm’s software product has limited quality issues and should release immediately as indicated by Corollary 1.

In part (ii) of Proposition 2, we analytically establish that when the firm’s development maturity is reasonably high (but not so high such that the bounds collapse to $L = H$), the firm should optimally set its release time close to $H$ under fairly broad conditions. Formally, we establish that the result applies when both network effects ($b$) and the impact of reduced quality on adoption ($c$) are low, but the product’s time horizon ($T$) is sufficiently long. Numerically, under a reduced level for $c$, Panel (c) of Figure 1 visually demonstrates how $t^*_0$ approaches $H$ as $\gamma$ increases even when $L$ and $H$ have considerable distance between these bounds. Panel (a) provides another example, demonstrating further breadth, for when the level of functionality in the product ($Y$) is higher.

Overall, the bounds $L$ and $H$ provide quite effective managerial advice with regard to software release time in the presence of error reporting. With Figure 1, we demonstrated that as a firm’s development processes become quite mature, it should release at $L = H$ (and early) which is analytically supported by Corollary 1. However, even if a firm is less mature, we can see that one of the two bounds will tend to be quite effective if used as a guidepost for release timing. Panels (b), (d), and (f) in Figure 1 all demonstrate that one of the bounds will generate a profit that is almost maximal. In panel (d), $H$ is a more effective bound, i.e. $\Pi(H) > \Pi(L)$ and $\Pi(H)$ is near $\Pi(t^*_0)$, because the firm can postpone release and continue bug detection and removal when there is a reduced quality effect on adoption. Specifically, it can bring to a market better product, hence incurring less goodwill costs, and still generate fast adoption upon release because of the reduced quality effect. In panel (f), $t^*_0 = L$ is precisely optimal for low software maturity, but even for higher maturity, releasing at $L$ will induce approximately maximal profits. Panel (b) is both interesting and representative of a large portion of the parameter space. Here, we see that releasing at $L$ generates near maximal profits for low maturity, while releasing at $H$ does the same for higher maturity. In other words, using the more appropriate bound for the right situation will go a long way. In this particular case, if a product has a lot of functionality and is developed by a
firm with low software maturity, then it needs to release its product to the market early in order to harness error reporting contributions from users and improve quality. On the other hand, when it uses more mature development processes, it has the ability to improve quality for a bit longer while avoiding goodwill costs. Finally, when it is extremely mature, there is no point in waiting with an already high quality software product, and it should release immediately as evidenced by the bounds collapsing.

Although the bounds generally perform well, in certain portions of the parameter space, numerical optimization of \( t_0^* \) is necessitated because neither bound performs near maximal. In Figure 2, we varied each cost parameter: testing \( (C_T) \), fixing \( (C_F) \), processing of user reported defects \( (C_{Pu}) \), and goodwill costs \( (C_G) \), from 0% to 300% of a benchmark level. We used dots to denote that the \( L \) bound performed better and lines with no marker when the \( H \) bound yielded higher profits. In panel (b), we demonstrate that under high maturity \( (\gamma = 0.85) \), the better bound is excellent, staying within 1% of the optimal profits. For \( C_T \), \( C_F \), and \( C_{Pu} \), \( H \) performs better. However, for \( C_G \), when goodwill costs are low, there is less of a downside of releasing at \( L \), in which case this bound is preferred to \( H \) because of discounting. However, as is presented in panel (a), when a firm has lower maturity \( (\gamma = 0.45) \), setting release time to either bound can significantly underperform \( t_0^* \) when goodwill costs are high.

Figure 2: The other parameter values are: \( a = 4, b = 0.5, c = 15, m(Y) = 30(1 - e^{-0.42Y}), T = 12, \alpha = 0.25, \beta = 0.05, \theta = 0.3, \theta_u(Y) = 0.1Y^{-0.7}, \hat{C}_T = 0.7, \hat{C}_F = 0.7, \hat{C}_{Pu} = 0.02, C_{P,f} = 0.01, \hat{C}_G = 0.4, C_D = 0.5, p = 4, \) and \( w = 0.46. \)
4.1.2 User Contributions and Release Timing

Next, we seek to provide some insights into the sensitivity of the optimal release time with respect to user contributions. First, we examine how an increase in the proportion of users who agree to participate in error reporting will affect the firm’s optimal release time. Second, we study how differences in the way functionality affects the users’ contribution to detection rates in turn impact the firm’s release time decision problem.

**Proposition 3** \( t^*_0 \) is weakly decreasing in \( \alpha \) if \( C_T \geq (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-L(\gamma)\theta} \) and weakly increasing in \( \alpha \) if \( C_T \leq (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-T\theta} \).

Proposition 3 demonstrates that the net effect of the interaction between the optimal release time and the level of error reporting is heavily determined by process maturity and the relative magnitude of both testing costs and the additional cost of processing user-generated error reports. When this additional processing cost is relatively low and process maturity is high, then, as user participation in error reporting increases, the firm has incentives to release earlier to offload more of the bug detection effort to consumers. However, if the additional processing cost of user reports is a bit higher, and the firm is less mature in its development processes, then the firm has incentives to limit the extent to which it harnesses consumer error reporting. Thus, if user participation in error reporting increases, a firm may strategically delay release, detect more bugs internally, and release a more polished product on the market, thus limiting user contributions.

The impact of product functionality \( Y \) on the firm’s profits and its release time decision is complex, hinging on how a change in functionality will alter the overall user contribution to the detection rate \( \theta_u(Y) \). To simplify the problem and more clearly focus on the most relevant trade-offs, in the following, we will hold the market potential constant at a constant level \( m \). Implicitly, we are zeroing in on a region where the market potential is not too elastic with respect to functionality. Wittingly, the scope of our paper takes functionality as given and focuses on the release time and pricing decisions. However, in consideration of the fact that functionality is indeed a choice in a prior decision problem, here we assume that the level of functionality is sensibly chosen to be in a range where \( m(Y) \) becomes inelastic at which point we can explore the effect of marginal changes on release timing.

**Proposition 4** Suppose the market potential for a software product is fixed. Then, there exist bounds \( \bar{\kappa} \), \( \kappa \), and \( \bar{\mu} \) such that we have the following comparative statics relating the optimal release time with the degree of product functionality:

(i) When the developing firm has highly mature development processes and the additional cost of processing user reported bugs is low, then
(a) If the impact of increased functionality on the user community’s contribution to detection rate is weak (i.e., \(Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < \bar{\kappa}\)), then \(t_0^{*}\) is increasing in \(Y\);

(b) If the impact is strong (i.e., \(Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) > \kappa\)), then \(t_0^{*}\) is decreasing in \(Y\).

(ii) In contrast, when the developing firm uses less mature development processes and the additional cost of processing user reported bugs is high, then,

(a) If the impact of increased functionality on the user community’s contribution to detection rate is a loss (i.e., \(Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < \bar{\mu} < 0\)) and goodwill costs are low, then \(t_0^{*}\) is decreasing in \(Y\);

(b) If the impact is strong (i.e., \(Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) > 0\)), then \(t_0^{*}\) is increasing in \(Y\).

For brevity, the explicit forms of \(\bar{\kappa}\), \(\kappa\), and \(\bar{\mu}\) are omitted from the main text and presented in the Appendix. Proposition 4 examines more comparative statics of the optimal release time, focusing on the impact of software functionality. An important measure that helps determine the aggregate effect of a change in functionality on release time is the degree to which this change affects the total detection rate contributed by users across all modules, i.e., \(Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y)\). When the impact on user contribution is weak (i.e., \(Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y)\) is small or negative), an increase in functionality leads to either a limited positive increase or a loss in total user detection rate. There are multiple effects in play; on one hand, the firm has reduced incentives to release its product early to the market because of the lower benefit associated with user contributions. Instead, the firm can shift its release time later to avoid the goodwill costs associated with early release. On the other hand, because there is a cost associated with processing user-reported defects, having less contributions from the user community may permit earlier release to engender faster adoption benefitting from network effects while incurring less additional processing costs.

Part (i.a) of Proposition 4 establishes that when a firm utilizes mature development processes and the cost of processing user-reported defects is small, then the optimal release time should be delayed with an increase in functionality. Referring to the trade-offs identified above, because the user contributions are weak, the firm can benefit from extending out the release time such that goodwill costs are not incurred while the firm continues testing on its own to improve the quality of its software by reducing defects. However, even though testing costs might be large, having higher software maturity will limit the firm’s exposure to these costs and permit a later release to ensure a better product is released to the market. Part (ii.a) of Proposition 4 examines the impact of a loss in user contributions as a result of increased functionality when the firm has lower software maturity and user error-report processing costs are higher. The idea here is that, beyond a certain point, increased complexity may impact the usability of a system and hence reduce the
chances of users spotting individual defects. Because there are a lot of flaws due to lower maturity, a decrease in user-reporting can be beneficial and permit earlier release without incurring as much total additional processing costs because of these reduced user contribution rates. In this case, releasing earlier can expedite adoption and be preferable as long as goodwill costs are limited.

When the impact on user contribution is strong (i.e., \( Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) \) is large), an increase in functionality leads to a strong boost to the total detection rate stemming from users. In this case, there is again a distinct trade-off. On one side, there are incentives to release the software earlier in order to benefit from the stronger user contributions. On the other hand, the firm also wants to limit both goodwill costs stemming from early release as well as potential additional costs associated with processing of user-reported defects. Part (i.b) of Proposition 4 establishes that when firms have higher development maturity, and correspondingly less bugs in their products, they can release earlier in order to benefit from the stronger user contributions as long as processing costs are not too high. High maturity and lower processing costs together limit the downside costs, making it optimal to decrease release time as a consequence to increased functionality. On the other hand, part (ii.b) of Proposition 4 formalizes the opposite effect: when a firm has lower software maturity, it is preferable to delay release to limit exposure to the additional processing and goodwill costs stemming from the higher bug density found in its product.

### 4.1.3 Beta Testing and Numerical Optimization of Detection Rates

In this section, we briefly examine how beta testing interacts with the firm’s release time decision. Because the focus of our paper centers on release timing and pricing in the context of user error reporting contributions, we abstract from initial software development decisions taking a software product as given (i.e., a release candidate is available at time zero). In the typical software release cycle, beta testing is a precursor to the release candidate stage, and therefore issues such as when to cease development (quality choice) and begin alpha/beta testing become more relevant. In that light, in this section, we do not intend to provide a comprehensive understanding of how to manage beta testing. Instead, we hope to provide insights on how the existence and degree of beta testing might impact a firm’s software release timing, which is a fundamental concern in our study.

We take a simple view of beta testing where the firm has the capability to harness a portion of the potential market \( N_B \in (0, m) \) to serve as testers of the product during the interval \([0, t_0]\) before release. In effect, beta testing helps the firm lower its own pre-release cost of testing by relying more on outside feedback while not incurring a significant cost of quality since the product has still yet to be released. At the same time, beta testing may push up the cost of processing user reported bugs by increasing the rate at which these detection reports arrive to the development team. In
the following proposition, we establish that increasing the extent of beta testing can both increase and decrease the optimal release time, depending on a firm’s process maturity.

**Proposition 5** When a software firm has highly mature development processes, it will tend to optimally delay its product release as the size of the beta testing group increases. However, when it has lower process maturity, it may find it optimal to release earlier in response to such an increase in the beta testing group.

Proposition 5 formalizes that when a firm with high process maturity produces a high quality product and there are not many defects in the code that need to be detected, it should delay its software release as the beta test size increases. When there are few defects in the code, the firm will not incur significant additional processing costs due to user reported bugs. However, it incurs testing costs at each moment while searching for these defects. As the size of the beta test increases, the firm has incentives to further reduce its own testing costs by delaying release and harnessing beta testers for a longer period of time because, once the product is released, it bears testing costs until adoption grows enough to generate sufficient detections stemming from user error reporting. On the other hand, when the firm has low software maturity, its product is characterized by a much higher defect density. In this case, because of the higher number of defects, it may incur higher additional processing costs from beta testing. As the size of the beta test increases, the firm has incentives to restrict the length of the beta test by releasing its product earlier, thereby incurring more testing costs but less processing costs as a result.

In the remainder of this section, we relax the simplifying assumption concerning a constant bug detection rate we made for tractability (see our discussion at the beginning of Section 4). In particular, we now permit the firm to optimize its level of contribution to the detection rate, $\theta_f(t)$, over the entire time horizon. Specifically, we will numerically study the following relaxed problem:

$$\max_{t_0 \in [L(\gamma), T]} \Pi(t_0, \theta_f(\cdot)) \quad \text{s.t. } \theta_f(\cdot) \leq \bar{\theta}_f$$

(17)

In this case, the total detection rate over time, $\theta(t)$, will be nonhomogeneous as it is influenced by the solution of (17), $\theta^*_f(t)$. We consider a constrained optimization over $\theta_f$ since, realistically, firms are constrained in how much they can ramp up internal testing. Numerically, we will demonstrate that our analytic results and insights stemming from the simple model which we employ are robust when permitting optimization of a firm’s contribution to detection. As described in the model development, if the firm contributes $\theta_f(t)$, then it incurs a cost rate of $C_T \times \theta_f(t) \times Y$. Noting that the costs are linear in $\theta_f(t)$, the firm’s numerically-computed optimal contribution $\theta^*_f(t)$ tends to
follow a bang-bang control policy.

First, we begin our numerical study by examining how user contributions affect optimal release timing. Allowing the firm to optimize its contribution to error detection, we illustrate in Figure 3 how the essence of Proposition 3 and relevant insights are robust to this relaxation. We plot the firm’s optimal release time $t^*_0$ as a function of the proportion of users who contribute to error reporting ($\alpha$) for two sets of parameters. Curve A represents a situation where testing costs are higher, processing costs are lower, and the firm has higher process maturity in software development. Curve B represents a contrasting scenario where testing costs are lower, processing costs are higher, and the firm exhibits less process maturity. The conditions of Proposition 3 (which clearly would differ when the firm optimizes its detection contribution) suggest that the optimal release time would decrease in $\alpha$ in the former case and increase in $\alpha$ in the latter case. Figure 3 demonstrates that the nature of the comparative statics we analytically characterized in Proposition 3 are unchanged even if we permit the firm to optimize $\theta_f^*(t)$. One might naturally ask then how robust is Figure 3; that is, does the numerical support also exhibit a broad parameter region where it is consistent? To answer this question, we performed a sensitivity analysis over the entire
Figure 4: How the optimal release time changes in software functionality/complexity, when the firm optimizes its own contribution level to detection. The common parameter values are: \(a = 10\), \(b = 0.5\), \(c = 5\), \(T = 1.5\), \(\alpha = 0.25\), \(\beta = 0.05\), \(C_F = 0.7\), \(C_D = 0.5\), \(C_P,J = 0.01\), \(\theta_f = 3\), \(m = 30\), \(\theta_u(Y) = 10Y^{-2}\), \(\sigma(\gamma) = 2(1 - \gamma)\), \(w(p) = 0.5e^{-0.02p}\), \(p = 1.7\), and \(\xi = 2\). For curve A, the specific parameter values are \(\hat{C}_T = 1.5\), \(\hat{C}_{P,u} = 0.02\), \(\hat{C}_G = 2\), and \(\hat{\gamma} = 0.7\). For curve B, the specific parameter values are \(C_T = 0.01\), \(C_{P,u} = 2\), \(C_G = 0.001\), and \(\gamma = 0.2\).

Next, we reexamine Proposition 4 in the context of (17) where the firm optimizes its contribution to detection. Figure 4 illustrates two examples satisfying parts (i.a) and (ii.a) of Proposition 4, which is to say that our choice of \(\theta_u(Y)\) satisfies \(Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < 0\). In panel (a), Curve A again represents a situation where both testing and goodwill costs are higher, processing costs are lower, and the firm has higher process maturity in software development. As indicated in part (i.a) of Proposition 4, Curve A also confirms that the optimal release time increases in software functionality under such conditions. When the firm can optimize its contribution, it will optimally maximize its effort up to its testing contraint until a point in time at which it ceases its own testing and relies on the contributions trickling in from users, which then grows as adoption takes off. This structure can be seen in panel (b) of Figure 4. The intuition as to why \(t^*_0\) increases in \(Y\) remains consistent with the discussion provided after Proposition 4; this is unchanged when the simplifying assumption that \(\theta(t)\) is kept to a constant is relaxed. Curve B, also illustrated in Figure 4, just as before represents where testing and goodwill costs are lower, processing costs are higher, and the firm exhibits less process maturity. Consistent with part (ii.a) of Proposition 4, we expect that \(t^*_0\) decreases in \(Y\) as previously discussed. Again, similarly, panel (c) of Figure 4 illustrates that the
Figure 5: How the optimal release time changes in the extent of beta testing, when the firm optimizes its own contribution level to detection. The parameter values are: $a = 4.8$, $b = 0.5$, $c = 7$, $T = 6$, $\alpha = 0.2$, $\beta = 0.05$, $C_T = 3.5$, $C_F = 1.7$, $C_G = 6.8$, $C_D = 1$, $C_{P,f} = 0.01$, $C_{P,u} = 1.9$, $\theta_f = 0.9$, $m(Y) = 40(1 - e^{-0.42Y})$, $\theta_u(Y) = 0.2Y^{-0.5}$, $\sigma(\gamma) = 2(1 - \gamma)$, $w(p) = 0.5e^{-0.02p}$, $p = 5.2$, and $\xi = 2$.

firm will switch from a high effort level to relying on users for detection shortly after releasing its product. As would be expected, the two curves illustrated in Figure 4 are also robust to a wide parameter region as in the case of Figure 3.

We return to the investigation of beta testing at the beginning of this section. Proposition 5 establishes that firms with higher process maturity will tend to delay the time of release with an increase in beta testing. However, firms with lower maturity will tend to release earlier with an increase in the size of the beta test. As can be seen in Figure 5, when the firm can optimize its level of contribution to detection, the results given in Proposition 5 for low and high maturity are fairly robust when the size of the beta test is not too high. In particular, under high software maturity (i.e., $\hat{\gamma} = 0.85$), the optimal release time is increasing in the size of the beta test as long as the proportion of users involved in beta testing is less than 90%. Under low software maturity (i.e., $\gamma = 0.35$), the optimal release time is generally decreasing in the size of the beta test. In most cases, beta tests do not involve significantly large percentages of the potential user base. However, when the size of the beta test becomes really large, Proposition 5 may become less applicable. In particular for the case of a higher process maturity firm, when the size of the beta test becomes large, the detection rate stemming from beta testers can be much larger for a prolonged pre-release
period which then permits the firm to release earlier because more defects would have been found in a shorter period of time. This is illustrated in Figure 5 for the right-hand side of the curve labeled $\hat{\gamma} = 0.85$.

### 4.2 Optimal Pricing

In the previous section, we examined a software firm’s optimal release timing and profitability when it utilizes release time as its primary lever to manage adoption and error reporting. Next, we explore a setting where a software firm also has some pricing power, optimally selecting a single price to charge throughout the selling horizon of the product. There are many instances where software firms prefer to set a price at an optimal level and keep it to a large extent at that level for the entire selling horizon. For example, in the past several years, AutoCAD suite and DivX video software bundle have been priced at $3995 and $19.99 respectively for a perpetual license, whereas WinEdt’s non-commercial single user license has been priced at a steady $40 over the last decade.

In this section in a stylized context of simple pricing, our aim is to better understand what role each lever plays to regulate adoption and harness the benefits of error reporting. We explore the general variable pricing as an extension in Section 5. We denote the optimal price chosen by the firm with $p^*$ which together with the optimal release time satisfy

$$ (t_0^*, p^*) = \arg\max_{t_0 \in [L(\gamma), T(Y)], p} \Pi(t_0, p). $$

In the following proposition, we demonstrate that even limited pricing power in a software firm’s respective market sometimes enables it to pursue early release strategies to more quickly harness consumer error reporting while strategically using price to shape demand.

**Proposition 6** Consider a software product which is either (i) produced by a firm with high maturity and targets a market with low error detection contribution ($\alpha \theta_u(Y)$) from users, or (ii) produced by a firm with intermediate process maturity, targets a market with low error detection contribution from users, and contains a lower level of functionality, or (iii) regardless of the developer’s process maturity, exhibits low processing cost for user-generated error reports, targets a market with high error detection contribution from users and low goodwill costs. Then, the software should be optimally released as soon as its quality is sufficient to induce adoption, i.e., $t_0^* = L(\gamma)$.

Parts (i) and (iii) of Proposition 6 together demonstrate that a firm does not have incentives to delay the release of a product of very high quality since costs are very small in this case and

---

7A technical formulation of Proposition 6 can be found in the Appendix.
adoption can be strong early on. Having the software on the market for the maximum feasible sales window dominates all other strategies. Moreover, part (iii) also reveals that an earlier release might be independent of process maturity (or, equivalently, product quality) if the goodwill cost is small and user detection contribution is sufficiently high. This would correspond to software markets with reduced or no competition where users have very limited capabilities to transfer any losses onto the firm and voluntarily contribute to quality improvement since it indirectly benefits all of them. Part (ii) captures an intermediate scenario where the product has an intermediate quality but low functionality and users are more reluctant to contribute to error reporting. As long as quality is not too low and functionality is limited, adoption can be robust from the beginning. Given the low user contribution to debugging, the firm commits to put in strong testing efforts throughout the product lifetime anyways, but the overall internal testing cost is very manageable given the low complexity of the product. Moreover, low functionality translates into low bug count, which also diminishes the impact of goodwill penalties and processing costs for user reports. Overall, the impact of shifting $t_0$ is relatively low on the cost side. Thus, given that early revenue is more valuable than later revenue, the advantages of selling early overcome other potential drawbacks.

When a firm releases its software at the earliest time possible, i.e. $t^*_0 = L(\gamma)$, it uses a two-pronged strategy. First, immediate release jump-starts adoption and permits error reporting feedback earlier in the software’s life cycle. Second, the firm finds it profitable to price the software (and, implicitly, shape adoption) in a manner that can help prevent overexposure to lower initial quality. In the remaining part of this section we explore the properties of optimal price in such scenarios. First, let us explore the link between price and the rate $\alpha$ at which users report bugs.

When the software is produced by a highly mature firm, it has few bugs. Given that the firm is performing extensive testing on the entire code (including error-free portions) and there are very few flaws remaining, if it can efficiently harness user contribution in detecting those, it has strong incentives to reduce the internal testing. This is done by shifting price in a balanced way so that the installed base leads to a significant reduction in testing costs without severely impacting revenue. The higher the user error reporting rate, the lower the optimal price is because a higher resulting installed base provides the firm with the ability to run a leaner internal testing by outsourcing detection costs on the user base without compromising on quality or profitability. However, when quality is not so high, if the firm releases early, it has the potential to incur a lot of goodwill costs and processing costs associated with user reports. When $\alpha$ increases, processing costs increase as well. Consistent with traditional monopolistic settings, an increase in average costs per user is compensated through a higher price which allows the firm to retain high margins by serving in the beginning a reduced mass of adopters for whom the product is mission-critical even under
the markup and lower quality and inducing a delay for the other adopters until quality further improves. Same argument applies for an increase in goodwill or error report processing costs. In contrast, when the internal testing cost increases, a larger installed base helps the firm offload to a greater extent testing efforts onto the consumers, and adoption incentives are provided through a lower price. These insights are summarized in the following proposition.

**Proposition 7** The optimal price satisfies the following comparative statics:

(i) $p^*$ decreases in $\alpha$ for software produced by a high maturity firm;

(ii) $p^*$ increases in $\alpha$ for software produced by a lower maturity firm (i.e., $\gamma$ satisfying $\frac{C_Te^{\theta_T}}{C_{P,u} - C_{P,f}} < \sigma(\gamma) < z_1$), that has lower functionality and generates low error detection rates from users;

(iii) $p^*$ decreases in $C_T$ and increases in $C_G$ and $C_{P,u} - C_{P,f}$ for both sets of conditions specified in (i) and (ii).

Next, we explore how pricing varies with functionality. In order to better illustrate the tradeoffs, we consider again a simple setting where the market potential is constant. Note that functionality factors in a complex way in both testing and processing costs. In the case of the former, it impacts user contribution to debugging ($\alpha\theta_u(Y)$) and the scope of the code (since the firm performs testing routines on the entire code). Processing costs are again influenced by the user contribution to debugging, and functionality is positively correlated with the inflow of reports of new bugs since it influences the total bug count. Thus, quantity $Y\theta_u(Y)$ captures the impact of functionality on the magnitude of savings in testing expenses net of processing costs for user generated reports. In turn, goodwill costs are impacted in a linear way by functionality. As functionality increases, the change in $Y\theta_u(Y)$ (i.e., $\frac{\partial Y\theta_u(Y)}{\partial Y} = Y\frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y)$) dictates whether savings in testing costs compensate for the increase in processing and goodwill costs. In that sense, the increase in bugs in association with increased functionality is considered in tandem with how users detect bugs in more complex software. Depending on the outcome, the firm may want to target a smaller or a larger consumer base and will do so through discounts or markups. Based on these dynamics, in the next result we extend our understanding of how a software firm should adjust its optimal price with regard to changes in software functionality (complexity) dependent on its software process maturity and the nature of end user contributions to error reporting.

**Proposition 8** Suppose the market potential for a software product is fixed. Then, we have the following comparative statics relating its optimal price with its degree of product functionality:

(i) When the developing firm has highly mature development processes,
Figure 6: Shape of optimal price paths for various process maturity and functionality scenarios. Parameters for Curve A: $\gamma = 0.95$, $\theta_u(Y) = 0.01Y^{-1}$, $C_T = 3.7$, $C_{P,u} = 0.5$, $C_G = 0.8$. Parameters for Curve B: $\gamma = 0.73$, $\theta_u(Y) = Y - 8 + 16Y^{-1}$, $C_T = 0.1$, $C_{P,u} = 1.0$, $C_G = 0.1$. Common parameters: $a = 6$, $b = 0.5$, $c = 10$, $\sigma(\gamma) = 2(1 - \gamma)$, $w(p) = e^{-0.1p}$, $T = 4$, $\theta = 0.25$, $\xi = 2$, $\alpha = 0.09$, $\beta = 0.05$, $m = 10$, $C_P = 0.05$, $C_F = 0.05$, $C_{P,f} = 0.01$. We consider $Y \in [0.5, 25]$ in order for $\theta_u(Y)$ to remain upper bounded.

(a) If the impact of increased functionality on the user community’s contribution to detection rate is weak (i.e., $Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < \bar{\kappa}$), then $p^*$ is increasing in $Y$;

(b) If the impact is strong (i.e., $Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) > \kappa$), then $p^*$ is decreasing in $Y$.

(ii) In contrast, when the developing firm has less mature development processes, functionality level is relatively low, and processing costs are relatively high,

(a) If the impact of increased functionality on the user community’s contribution to detection rate is a loss (i.e., $Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < \bar{\mu} < 0$), then $p^*$ is decreasing in $Y$;

(b) If the impact is strong (i.e., $Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) \geq 0$), then $p^*$ is increasing in $Y$.

Quantities $\bar{\kappa}$, $\kappa$, and $\bar{\mu}$ are the same as in Proposition 4 and they are described in the Appendix.

Proposition 8 explores price monotonicity under conditions which lead to an optimal release at the earliest time possible (as per Proposition 6). Similar intuitions as in Proposition 4 apply here as well in regards to the cost tradeoffs and the firm’s preference for a larger or smaller installed base. However, in this case the firm uses price as the lever to shape adoption. When the firm has high maturity (scenario (i) of Proposition 8), an increase in functionality is associated with relatively few new bugs. Moreover, when quality is high, savings in testing costs associated with a higher installed base dominate increasing processing costs. Nevertheless, when the impact of increased
functionality on user detection rate is very low (or negative), the increase in goodwill cost due to those few new resident flaws dominates the fluctuations in both processing and testing costs. In such a case, the firm prefers to assume a larger portion of the testing costs and serve a smaller installed base under higher margins. This scenario is depicted in the LHS part of curve $p^*_A$ in panel (b) of Figure 6. Alternatively, if consumers’ usage rate for more complex software is considerably higher and, as a result their detection rates per bug increase (or do not decrease significantly), then the firm can offload a lot of testing effort onto the users which overcompensates for any increase in processing or goodwill cost. In such a case, the firm marks down the price to boost adoption, as illustrated in the RHS of curve $p^*_A$.

On the other hand, when the process maturity is lower, falling into an intermediate range (i.e., $\frac{C_{tp}}{C_{tpu}} < \sigma(\gamma) < z_1$), the effect of how $Y\theta_u(Y)$ responds to a change in functionality has a much different impact on the optimal price. Under such a scenario, testing savings due to installed base contributions are dominated by the loss associated with processing user-generated reports. Also, an increase in functionality leads to an increase in goodwill costs. However, if such an increase severely inhibits user error detection, then the overall loss due to processing costs net of savings on the testing side is actually diminishing. The firm might incur higher testing costs and higher goodwill costs but it will incur far less processing costs, and the aggregate savings are more pronounced when there are more users. This is illustrated in LHS of curve $p^*_B$ in panel (b) of Figure 6, for very small values of $Y$. Nevertheless, when an increase in functionality actually leads to a strong impact on user contribution to detection, then processing costs increase and the firm has incentives to charge higher prices in order to get into the low-volume-high-margin region, thus limiting exposure to both processing and goodwill costs, as illustrated in the middle region of curve $p^*_B$, for $Y > 3$.

In scenario (ii) of Proposition 8, given low functionality, the firm releases the software at the earliest time possible and price is sufficient to efficiently shape demand to yield maximum profitability. However, when software functionality is very high, quality is low, and software usage increases significantly with functionality (hence user contributions increase significantly with software complexity), the ability of a firm to set a fixed price over the software product’s lifetime may be insufficient to induce the most profitable adoption path. In this case, the firm may strategically delay its release timing to work in conjunction with its price to better manage adoption. This can be seen in the RHS of curve $p^*_B$ in panel (b) of Figure 6. When $Y > 15.45$, then $t^*_0$ moves away from $L_B$ and actually starts increasing, which relaxes substantially the pressure on the optimal price, as exhibited by the price curve diverging below the dashed line (corresponding to optimal price under earliest possible release). In such cases, the software is released in the market at a better quality.
level and average costs per user due to goodwill penalties and processing of user-generated reports are lower, which in turn may even lead to a decrease in price.

5 Extension - Increased Price Control

In the following extension, we study how a firm with increased price control will shape its price trajectory to affect adoption in our context of user error-reporting contributions. From a bird’s eye perspective, we aim to provide insight into how a firm adjusts its adoption management strategy when it has additional pricing power. The software firm can choose a price function \( p(t) \) which details how its product’s price changes over time. Often, when a product gets closer to discontinuation, its price is lowered. However, in various other situations, software price can also be increasing. Indeed, in the software industry, we see firms adjusting price in both directions. For example, in June 2008, Oracle increased the price of its E-Business Suite applications and its licence (per CPU) for database software by 15-20% (Kanaracus 2008). Or, more recently, Microsoft reduced prices on Windows 7 by 35% (Bertolucci 2010). In this section, we explore what type of price paths may be preferable for firms of different process maturities that operate in different segments.

In addition to those listed in section 3, we assume that an additional regularity condition on the price response function is satisfied: \( \lim_{p \to -\infty} w(p) = \infty \). This condition states that if the firm sufficiently subsidizes the product, then adoption is extremely fast with very little market potential remaining before full adoption occurs. Given heterogeneity in the user base, sometimes a small subsidy can be critical to ensuring willing users who make up the market potential can overcome learning costs, complementary hardware costs, and other barriers to product adoption. We often see firms seeding some portions of the market. For example, Microsoft offers (for free) professional grade software to students worldwide through its Dreamspark program. IBM has a similar program (IBM Academic Initiative) that targets researchers and faculty. On Apple’s iTunes App Store, some applications will go through a short period where they are offered for free, in a bid to boost installed bases. As a recent example of covering adoption costs, Microsoft paid a quarter of a million dollars to University of Nebraska to incentivize a switch to Office 365 from IBM Lotus Notes (McDougall 2011). In our model, \( p(t) < 0 \) simply indicates the firm is offering for free the product and covering (partially or completely) the associated adoption costs at time \( t \), and its impact on demand is still given by (4) through \( w(p(t)) \). While such a strategy cannot yield positive profits under a constant price scenario as in Section 4.1 and Section 4.2, when firms have greater control of price over time, we should not restrict their ability to strategically subsidize. In particular, a firm may find it

\[ \text{Any subsidy must be structured such that only consumers who intend to use the software are provided incentives to consume.} \]
optimal to subsidize adoption early on to boost network effects and error reporting contributions, hence making greater profits over the entire selling horizon as a result.

In the previous section, we demonstrated that when a firm has limited control over price, release time plays a significant role in managing adoption toward profitability. In some cases, the firm should purposely delay the software release and continue to improve it internally for an extended period of time. Specifically, it may be the case that a fixed price level, even if chosen by the firm, does not provide sufficient adoption shaping ability, hence profit maximization requires release timing adjustments. For the increased price control case we study in this section, the firm’s optimal price path and release time must satisfy

\[
(t^*_0, p^*(t)) = \arg\max_{t_0 \in [L(\gamma), T], p(t)} \Pi(t_0, p(t)).
\]

(19)

When the firm has sufficient control of price, we demonstrate that releasing at the earliest time possible for its given level of software maturity is optimal. It can then rely on price to influence customer error reporting and manage adoption as seen in the following proposition.

**Proposition 9** When a firm optimizes its product’s entire price path \( p^*(t) \), it should release the product at \( t^*_0 = L(\gamma) \).

Proposition 9 formally establishes that full price control renders any strategic delaying of a product release as redundant. Technically, if one considers the feasible strategy \((t_0, p(t))\), then the adoption it induces can be replicated by an alternative strategy \((\hat{t}_0, \hat{p}(t))\) satisfying \( \hat{t}_0 = L(\gamma) \), \( \hat{p}(t) = \infty \) for all \( t \in [L(\gamma), t_0] \), and \( \hat{p}(t) = p(t) \) otherwise. By Propositions 1, 6, and 9, we see that the extent to which a firm can adjust price over time helps to determine the point at which it should release its software. Although it may be unlikely that a firm can effectively change its price continuously, the insights stemming from our results in this setting bring forth an important implication: As harnessing consumer error reporting becomes standard development practice, we expect that software firms with greater pricing power in their respective markets will also release their products earlier.\(^9\)

Given a firm should release at a time \( L(\gamma) \) that depends on its level of process maturity, for the remainder of this section we focus our analysis on how firms should adjust the price path over time to manage adoption, as further driven by product functionality and process maturity. Accordingly,

---

\(^9\)Proposition 9 does not state that low quality products should be released immediately at time zero, the beginning of the release candidate stage. More clearly, it states that a software firm with development maturity \( \gamma \) should release its product at \( L(\gamma) \), the earliest time that a firm with maturity \( \gamma \) can develop sufficient quality that adoption by consumers will commence. This threshold depends on software process maturity because it is positively correlated with quality, as well as demand-side primitives such as intrinsic benefits from software, strength of network effects, and sensitivity to flaws in a certain software category. As can be seen in Figure 1, firms with higher maturity will clearly release earlier than those with lower process maturity.
in the remaining part of this section, we derive structural properties of \( p^*(t) \). For convenience, we first define threshold value \( z_2 = \frac{a-b}{\Gamma(T)} \) and consider \( z_1 \) as defined in Proposition 2.

**Proposition 10** Suppose a firm uses mature development practices, i.e. \( z_2 > \sigma(\gamma) \), and offers a software product with a low user error reporting rate (perhaps by not offering automated error reporting tools) such that \( C_G \Gamma(T) > \alpha \theta u(Y) \left[ \frac{C_G}{\sigma(\gamma)} - (C_{P,u} - C_{P,f})e^{-\theta T} \right] \) is satisfied. Then,

(i) There exists a time \( t_1 \in [L(\gamma), T] \) at which the firm optimally transitions from higher prices to a price-skimming strategy throughout the remainder of the software selling horizon.

(ii) Further, suppose the firm’s process maturity is sufficiently high, i.e. \( z_1 > \sigma(\gamma) \). Then \( L(\gamma) = 0 \) and:

(a) Although it may use penetration pricing, the firm should never subsidize adoption at any time.

(b) In the absence of discounting, the firm should avoid penetration pricing altogether. Rather, the use of a price-skimming strategy during the entire life cycle becomes optimal. That is, \( \partial p^*(t)/\partial t < 0 \) for all \( t \in [0, T] \).

Proposition 10 provides a characterization of the optimal price path for firms that have mature development processes and also bring to market a product that induces a low user error reporting rate. In this case, we find that firms should implement higher prices during the earlier portion of the sales horizon and decrease them toward its product discontinuation date, i.e., use a markdown strategy toward the end. A product with a given level of functionality includes fewer software flaws if it is developed by a highly mature firm in comparison to a less mature one, as seen in Table 1. However, if the software also has a high level of functionality, there are more opportunities for defects to occur. Further, when error reporting is low, the benefits of consumer feedback are difficult to harness. As a consequence, the firm is exposed to potentially high goodwill penalties early in the adoption cycle, when there are more bugs remaining in the code, and these penalties are exacerbated by a large installed base. Thus, very fast adoption in the beginning is suboptimal for the firm.\(^{10}\) As a result, there is no need to subsidize adoption, as shown in part (ii.a) of the proposition. However, depending on the price elasticity of demand and adoption speed primitives, the firm may offer lower prices to provide additional incentives for users to adopt early which, in turn, generates adoption momentum through network effects.

Parts (i) and (ii.b) of Proposition 10 together establish that one critical factor determining whether or not early penetration pricing is utilized is the discount rate. When future cash flows are

\(^{10}\)Note that high maturity actually increases the speed of adoption as \( \frac{B(t)}{\Gamma(t)} = \sigma(\gamma) \Gamma(t) \).
Figure 7: Shape of optimal price paths for low and high levels of process maturity and discounting, under full price control. These levels are given by $\gamma_L = 0.45$, $\gamma_H = 0.9$, $\beta_L = 0$ and $\beta_H = 0.10$. The other parameter values are: $a = 10$, $b = 9$, $c = 20$, $Y = 20$, $m(Y) = 3Y$, $T = 7$, $\alpha = 0.10$, $\theta = 0.05$, $\xi = 1$, $\theta_u(Y) = 0.3 \times Y^{-4}$, $C_T = 6$, $C_{P,f} = 0.05$, $C_{P,u} = 0.2$, $C_G = 0.015$, $C_F = 0.05$, $C_D = 0.04$, and $w(p) = 0.80e^{-0.5p}$.

not as heavily discounted, the firm has less incentive to offer low prices early on to induce faster adoption. In fact, when cash flows are not discounted at all, part (ii.b) of Proposition 10 formally establishes that the optimal price path is simply decreasing in time. This optimal price path shape is also illustrated in Figure 7 with the curve labeled A. In contrast, with sufficient discounting, the optimal price path is likely to have a inverted U-shape as illustrated with curve B in Figure 7; here, lower initial prices help expedite adoption. As time passes, the quality of the product improves through error reporting and becomes further valued due to the positive network effects. Hence, the firm optimally increases its price to higher levels to reap greater rewards during the intermediate portion of the sales horizon. As time nears the end of the sales horizon, part (i) of Proposition 10 establishes that the firm adjusts price downward to capture remaining users before discontinuation.

As depicted with curve B in Figure 7, a firm with high software process maturity may use penetration pricing early on although, as part (ii.a) of Proposition 10 establishes, it never subsidizes adoption. In contrast, when a firm offers a product that generates low user error reporting rates and uses software development processes that yield a lower rating on measures such as the Capability Maturity Model, while it may still use price skimming towards the end of product life cycle (as established in part (i) of Proposition 10), it may find it optimal to subsidize adoption right at release. With poor software development practices, there are many defects in the product and few
users want to adopt early in the life cycle. Instead, users naturally prefer to wait for the firm to spend more time debugging the software and for its quality to rise. By subsidizing early into the release, the firm can induce some of these users to adopt immediately, contribute to quality through error reporting, and boost the rate of adoption via positive network effects. Hence, a more pronounced, inverted U-shaped pricing path with initial subsidization can be an optimal strategy as depicted by pricing path C in Figure 7. Notably, this type of path hinges on the magnitude of goodwill costs being relatively low. If goodwill costs are too high, then combined with low process maturity, there are increased incentives for the firm to restrict early adoption. In this case, the firm may pursue a strategy where it throttles adoption in a coordinated manner with its efforts to improve quality over time. Thus, a decreasing price path can be optimal when a firm with poor development practices offers a high functionality product but is subject to large goodwill costs. Note that all optimal price trajectories end at the same point. It can be shown (Lemma A4 in the Appendix) that $p^*(T)$ represents the unique solution to the equation $pw'(p) + w(p) = 0$.

Next, we turn our attention to how high error reporting contributions as well as the interaction between testing costs and the length of the sales horizon affect a firm’s optimal price path.

**Proposition 11** Suppose a software product is in a class that generates relatively high user error reporting rates, i.e. $C_G < \alpha b_u(Y) \left[ \frac{C_T}{\sigma} - (C_{P,u} - C_{P,f}) \right]$.

(i) If a firm using less mature software development practices satisfying $z_1 < \sigma(\gamma)$ produces the software, then

(a) The employment of a price-skimming strategy throughout the entire software selling horizon is never optimal. Rather, there always exists a time interval where the optimal price path is increasing.

(b) When the discount rate is zero, the optimal price path increases immediately after release.

(ii) High testing costs induce the optimal price path to increase at the end of the product life cycle. However, low testing costs and a longer sales horizon may cause it to decrease instead at the end of the product life cycle.

In contrast to part (ii.b) of Proposition 10, part (i.a) of this proposition establishes that a software firm with low process maturity never implements a fully decreasing price path. Further, part (i.b) of Proposition 11 establishes that the firm can initially offer its product at a lower price and increase it from there. Since the firm has low process maturity (e.g., newer software firms that may have development processes consistent with level 1 in the Capability Maturity Model), there are many defects in the product that need to be addressed before the software is of higher quality. By offering
the product at lower prices early on, the firm accelerates adoption (which is negatively impacted by quality) and benefits from user error reporting without being exposed to high goodwill costs since \( C_G < \alpha \theta_u(Y) \left[ \frac{C_T}{\sigma} - (C_{P,u} - C_{P,f}) \right] \) is satisfied. Thus, in this case, stimulation of adoption and consequently consumer error reporting plays a critical role in the firm’s pricing decision and explains the initial shape of the optimal pricing path.

Part (ii) of Proposition 11 examines how testing costs and the length of the sales horizon together help determine whether a decreasing price strategy at the end of the horizon is profitable. High testing costs provide incentives for the firm to harness user feedback instead of internal testing resources. When these benefits outweigh goodwill penalties, the firm will boost early adoption via lower prices. Later, once the product quality improves, the benefit of faster adoption at low prices is small in comparison to the lost revenue. Thus, towards the end of the horizon, the firm increase prices. This effect tends to be more pronounced if the product life is short. However, different classes of software also have variable intrinsic life spans. For example, the time between major releases of productivity software is much shorter than that of geotechnical software due to the rate of hardware improvements and changes in technology. When the natural life span of a software product’s class is longer and testing costs are not prohibitively high, part (ii) of Proposition 11 establishes that the optimal pricing path decreases toward the end of the sales horizon. In this case, the horizon is sufficiently long that the software provider has time to increase quality to a high level while simultaneously increasing prices to extract surplus from higher value consumers and subsequently adopt a price skimming strategy to serve the remainder of the consumer market.

6 Concluding Remarks

With the recent, rapid penetration of broadband Internet access, software firms and their customers now interact at a higher frequency and complexity than before, especially after a product is released to the market. This modern structure enables firms to address software quality issues by leveraging both in-house developers and user/community resources. In this paper, we present a model of software adoption to develop an understanding of how firms, which vary in development characteristics such as software process maturity as well as product characteristics including class and functionality, should optimally adapt their release timing and pricing in order to harness the potential of customer error reporting. We study this issue in depth in both a setting where the firm has limited ability to control the price of its product as well as a setting where it can optimally set its product’s price.

In the former setting, we characterize relevant bounds on a firm’s software release time and
relate these bounds to the firm’s process maturity and its product’s level of functionality. We also explore in depth how consumer feedback can impact the benefit-cost dynamics and show that, contrary to conventional wisdom, under certain market conditions it can be preferable to delay release when either customer error reporting rates increase, software functionality decreases, or the size of a pre-release beta testing group increases. When the firm can control the price in a limited way, we demonstrate that higher process maturity or intermediate process maturity, low user error detection contribution, and low functionality tend to provide incentives for the software developer to release the product early and rely more on price to shape adoption. However, lower maturity firms may have incentives to delay release when they develop high functionality software. Moreover, we demonstrate that an increase in functionality may be associated with a price markdown for both higher and lower quality products. Consistent with the literature, for the analytical results in the paper we model the overall bug detection rate as being constant in order to maintain tractability. Nevertheless, we perform extensive numerical analysis optimizing firm’s detection efforts in Section 4.1.3 and show that our insights under the simplified model are quite robust to more general settings.

At the other end of the spectrum, where the firm has complete and continuous control of its price path, we first establish that it will release its product at the earliest moment and strictly rely on pricing to manage adoption. We find that firms that use mature software development practices and target markets with low levels of user engagement in error reporting should employ price skimming strategies toward the end of their product horizons. Moreover, penetration pricing can be optimally used early in a product’s life cycle if future revenues are sufficiently discounted. However, lower maturity developers whose software engenders a high level of user feedback are more likely to rely on penetration pricing since it is critical for them to leverage customer error reporting earlier in the product life cycle. Price-skimming can still be employed close to sales discontinuation when testing costs are sufficiently low and the product’s sales horizon is long. However, in contrast to firms with higher process maturity, a price-skimming strategy can never be optimal for the entire sales horizon.

For tractability, we do not explicitly model bug heterogeneity and capture it in implicit form in the detection and fixing processes, by assuming that some bugs will take longer to surface compared to others (or may not be found at all), and some detected (thus, known) flaws may take a longer time to get fixed compared to others. More realistically, compared to simpler bugs, the difficult ones can take more effort or longer time to identify and/or fix. Also, detection times are generally not memoryless, and detection times may not be independent since bugs can be related. Moreover, some bugs may be more harmful to the users while others can be quite innocuous, which
may induce the firm to prioritize the effort allocation in the patching process. While the stricter assumptions used in the paper still yield substantial insight into how firms manage release timing and pricing, an important next step can be to relax some of these assumptions to explore how a firm’s optimal decision directionally adjusts when incorporating bug heterogeneity. A second simplifying assumption we make is that all adopters uniformly use all the software product’s functions. However, our results and insights also do not critically depend on this assumption.

Since we employ a diffusion model, demand is aggregated at the market level and we do not layout the decision-making process of individual consumers. That being said, a subsequent extension to our work can be to take a utility-based approach and incorporate customer error reporting, release timing, pricing, and adoption. Although making analytical progress with a continuous time model in this type of approach will be difficult, such an extension can yield valuable additional insight into the way a software market is strategically grown. Finally, future studies might also examine competition, subscription-based revenue schemes, optimal liability sharing for bug-induced damages, and cannibalization from next-generation software releases.

References


Appendix for “The Influence of Software Process Maturity and Customer Error Reporting on Software Release and Pricing”

Derivation of adoption path \( N(t) \) for a given fixed price - equation (13). For \( t \geq t_0 \) we denote by \( F(t) = \frac{N(t)}{m(Y)} \) the market penetration at time \( t \) (i.e., the fraction of the total market potential that adopted before \( t \)). Let \( \bar{a} = aw(p) \), \( \bar{b} = bw(p) \), and \( \bar{c} = cw(p) \). From (4), given \( \bar{B} = \sigma(\gamma)Y \) we obtain the following hazard rate model when \( t \geq t_0 \):

\[
\frac{f(t)}{1 - F(t)} = \bar{a} + \bar{b}F(t) - \bar{c}\sigma(\gamma)\Gamma(t),
\]

(A.1)

where \( f(t|t_0, p, \gamma) = \frac{\partial F}{\partial t}(t|t_0, p, \gamma) \). This formulation is independent of market potential \( m(Y) \) and functionality \( Y \), which means the solution \( F \) will be independent of these parameters as well. Constraint \( t_0 \geq L(\gamma) \) ensures that \( F \) behaves like a cdf \( (F(t_0) = 0, F(\infty) = 1, \text{and } F \text{ is weakly increasing in } t \text{ over } [t_0, \infty)) \).

Denoting \( g(t) = \bar{a} - \bar{c}\sigma(\gamma)\Gamma(t) \), \( h(t) = \bar{b} - \bar{a} + \bar{c}\sigma(\gamma)\Gamma(t) \), \( \ell(t) = -\bar{b} \), we get the following Riccati partial differential equation:

\[
F'(t) = g(t) + h(t)F(t) + \ell(t)F^2(t),
\]

(A.2)

with boundary condition \( F(t_0) = 0 \). If we can find a particular solution \( F_1 \) to equation (A.2), then a more general solution is of the form \( F(t) = F_1(t) + \frac{1}{\nu(t)} \), where \( \nu(t) \) is a solution to the first order differential equation:

\[
\nu'(t) = -[h(t) + 2\ell(t)F_1(t)] \times \nu(t) - \ell(t).
\]

(A.3)

Analyzing equation (A.2), we see that \( F_1 \equiv 1 \) satisfies the required constraint. We get the following first order linear equation:

\[
\nu'(t) = [\bar{b} + \bar{a} - \bar{c}\sigma(\gamma)\Gamma(t)] \times \nu(t) + \bar{b}.
\]

(A.4)

Defining \( \rho(t|\theta, \xi, \gamma) \triangleq (b + a)t + \frac{c(\gamma)\beta t}{\xi^2 - \eta^2} \left( e^{-\frac{\theta t}{\xi^2}} - e^{-\frac{\xi t}{\xi^2}} \right) \), we get the following general solution to equation (A.4):

\[
\nu(t) = e^{w(p)\rho(t|\cdot)} \left( K + \bar{b} \int_{t_0}^{t} e^{-w(p)\rho(x|\cdot)} dx \right),
\]

(A.5)

where \( K \) is a constant that depends on \( t_0 \). Therefore, a solution for equation (A.2) will have the form:

\[
F(t) = \begin{cases} 
0 & \text{, if } t < t_0, \\
1 + \frac{1}{\nu(t)} & \text{, if } t \geq t_0.
\end{cases}
\]

(A.6)
Using boundary condition $F(t_0) = 0$ we obtain $K = -e^{-w(p)\rho(t_0)}$, and the solution follows. □

**Proof of Lemma 1.** (i) Since, as discussed in the proof of equation (13), $N(t) = m(Y)F(t)$, it is sufficient to show monotonicity of $F$ with respect to $t_0$ and $p$.

Monotonicity of $F$ with respect to $t_0$. We have $F(t|t_0, \cdot) = 1 + \frac{1}{w(t|t_0, \cdot)}$. Then, $\frac{\partial F}{\partial t_0}(t|t_0, \cdot) = -\frac{w'(t_0)}{w(t_0)}$. From equation (14), after computations and regrouping, we get:

$$\frac{\partial F}{\partial t_0}(t|t_0, \cdot) = w(p)e^{w(p)[\rho(t) - \rho(t_0)]}(a - c\sigma(\gamma)\Gamma(t_0)).$$  \hspace{1cm} (A.7)

Constraint $t_0 \geq L(\gamma)$ ensures that $a - c\sigma(\gamma)\Gamma(t_0) \geq 0$. Therefore, we see that $\frac{\partial F}{\partial t_0}(t|t_0, \cdot) \geq 0$. Consequently, $\frac{\partial F}{\partial t_0}(t|t_0, \cdot) < 0$.

Monotonicity of $F$ with respect to $p$. Let us consider $0 < p_1 < p_2$. Then we have $w(p_1) > w(p_2)$. At the origin $t_0$, we have $F(t_0|t_0, p_1, \cdot) = F(t_0|t_0, p_2, \cdot)$ and $f(t_0|t_0, p_1, \cdot) > f(t_0|t_0, p_2, \cdot)$. Therefore, in the vicinity of the origin, $F(t|t_0, p_1, \cdot)$ is strictly above $F(t|t_0, p_2, \cdot)$.

Suppose that at some point $\infty > t_1 > t_0$ the curves cross each other and $F(t_1|t_0, p_1, \cdot) = F(t_1|t_0, p_2, \cdot)$. Then, using either equation (4) or equation (A.1), we have $\frac{f(t_1|t_0, p_2, \cdot)}{f(t_1|t_0, p_1, \cdot)} = \frac{w(p_2)}{w(p_1)} > 1$. Thus, $f(t_1|t_0, p_1, \cdot) > f(t_1|t_0, p_2, \cdot)$. Define $G(t) = F(t|t_0, p_1, \cdot) - F(t|t_0, p_2, \cdot)$. Then we have $\frac{\partial G(t)}{\partial t} |_{t=t_1} > 0$. Therefore, due to continuity, there exists $\epsilon > 0$ such that $G(t)$ is strictly increasing on the interval $(t_1 - \epsilon, t_1 + \epsilon)$. We have $G(t_1) = 0, G(t) < 0$ for $t \in (t_1 - \epsilon, t_1)$, and $G(t) > 0$ for $t \in (t_1, t_1 + \epsilon)$. For that reason, if at any time later than $t_0$ the two graphs intersect, $F(\cdot|t_0, p_2, t_0)$ must cross $F(\cdot|t_0, p_1, t_0)$ from above. However, as argued above, $F(t|t_0, p_2, \cdot)$ starts below $F(t|t_0, p_1, \cdot)$ around origin. For that reason, if the two graphs ever intersect, the first time they do $F(t|t_0, p_2, \cdot)$ must cross $F(t|t_0, p_1, \cdot)$ from below. This is a contradiction.

(ii) When $Y$ is constant, similar to part (i), it is enough to show monotonicity of $F$.

Monotonicity of $F$ with respect to $\gamma$. Similar to the proof for monotonicity of $F$ with respect to $p$.

(iii) Trivial since $F(t)$ is independent of $Y$ and $m(Y)$ is increasing in $Y$. □

**Proof of Proposition 1.** Using (10) and (11), we immediately see that $D(t) - B + B(t) = \frac{\bar{B}\theta}{\xi - \sigma}(e^{-\theta t} - e^{-\xi t})$.

Using $\theta_f(t) = \theta - \alpha\theta_u(Y)N(t)$ and $N(t) = 0$ for all $t \leq t_0$, the four costs incurred by the firm can be written as:

$$\tilde{C}_T = \int_0^T C_T \times (\theta - \alpha\theta_u(Y)N(t)) \times Y \times e^{-\beta t} dt$$

$$= \frac{C_T Y \theta}{\beta} \left(1 - e^{-\beta T}\right) - C_T \theta_u(Y)\alpha Y \int_0^T N(t)e^{-\beta t} dt,$$  \hspace{1cm} (A.8)

$$\tilde{C}_P = \int_0^T \left( C_{P,j} \times \frac{\theta - \alpha\theta_u(Y)N(t)}{\theta} + C_{P,u} \times \frac{\alpha\theta_u(Y)N(t)}{\theta} \right) \times \bar{B}e^{-\theta t} \times e^{-\beta t} dt$$

$$= \int_0^T C_{P,j} \bar{B}e^{-(\beta + \theta)t} dt + \int_0^T (C_{P,u} - C_{P,j})\alpha\theta_u(Y)N(t)\bar{B}e^{-(\beta + \theta)t} dt$$

$$= \frac{C_{P,j} \bar{B} \left(1 - e^{-(\beta + \theta)T}\right)}{\theta + \beta} + (C_{P,u} - C_{P,j})\alpha\theta_u(Y)\bar{B} \int_0^T N(t)e^{-(\beta + \theta)t} dt,$$  \hspace{1cm} (A.9)
\[
\tilde{C}_F = \int_0^T (C_F + \theta C_D) \times \frac{B \theta}{\xi - \theta} (e^{-\beta t} - e^{-\xi t}) \times e^{-\beta t} dt,
\]
(A.10)

\[
\tilde{C}_G = \int_{t_0}^T C_G \times N(t) \times B(t) \times e^{-\beta t} dt.
\]
(A.11)

If the software firm releases the product at time \(t_0\), then, after integration by parts revenue can be rewritten as \(\int_{t_0}^T \partial N(t)p(t)e^{-\beta t} dt = pm(Y)F(T\mid \cdot)e^{-\beta T} + m(Y)p\beta \int_{t_0}^T F(t\mid \cdot)e^{-\beta t} dt\). Rearranging terms in equation (9), we obtain the following formula:

\[
\Pi(t_0, p) = pm(Y)F(T\mid \cdot)e^{-\beta T} - \frac{C_T Y \theta}{\beta} (1 - e^{-\beta T}) - \frac{C_p \beta \theta (1 - e^{-(\beta + \theta)T})}{\theta + \beta} - \tilde{C}_F + m(Y) \int_{t_0}^T e^{-\beta t} F(t\mid \cdot) \Phi(t) dt.
\]
(A.12)

We apply Leibniz integral rule\(^{11}\) and boundary condition \(F(t_0\mid t_0, \cdot) = 0\) in order to compute the derivative:

\[
\frac{\partial \Pi(t_0, \cdot)}{\partial t_0} = pm(Y)e^{-\beta T} \frac{\partial F}{\partial t_0}(t\mid t_0, \cdot) \bigg|_{t=T} + m(Y) \int_{t_0}^T e^{-\beta t} \frac{\partial F}{\partial t_0}(t\mid t_0, \cdot) \Phi(t) dt.
\]
(A.13)

We know from Lemma 1(i) that \(\frac{\partial F}{\partial t_0}(t\mid t_0, \cdot) \leq 0\) for all \(t \geq t_0\). Since \(B(t)\) is monotonically decreasing in \(t\), \(\Phi(t)\) is increasing in \(t\). Thus, when \(\Phi(t_0) \geq 0\), we have \(\Phi(t) \geq 0\), \(\forall t \in [t_0, T]\). Consequently, \(\Pi(t_0, \cdot)\) is decreasing in \(t_0\) whenever \(\Phi(t_0) \geq 0\).

We derive two conclusions here. First, if \(\Phi(L(\gamma)) \geq 0\), we know that \(\Pi(t_0, \cdot)\) is decreasing in \(t_0\) for any value \(t_0 \in [L(\gamma), T]\), which leads to \(t_0^* = L(\gamma) = H(Y, \gamma, p)\). Second, if \(\Phi(L(\gamma)) < 0\), then let \(\bar{t} > t_0\) represent the unique solution to equation \(\Phi(t) = 0\). If \(T \geq \bar{t}\), then, based on the above argument, \(\Pi(t_0, \cdot)\) is decreasing in \(t_0\) for any value \(t_0 \in [\bar{t}, T]\). The conclusion follows immediately. □

**Proof of Corollary 1.** Follows immediately from Proposition 1. □

**Lemma A1** Define \(z_1 \triangleq \frac{a - b}{c}\). The following hold true:

(i) If the firm exhibits high maturity \((\sigma(\gamma) \leq z_1)\) then \(\frac{\partial^2 F(t\mid t_0, \cdot)}{\partial t \partial t_0} \geq 0\) for all \(t_0 \in [L(\gamma), T]\) and all \(t \in [t_0, T]\).

(ii) If the firm exhibits low maturity \((\sigma(\gamma) > z_1)\) and the sales horizon \(T\) is sufficiently small, then \(\frac{\partial^2 F(t\mid t_0, \cdot)}{\partial t \partial t_0} \leq 0\) for all \(t_0 \in [L(\gamma), T]\) and all \(t \in [t_0, T]\).

**Proof.** (i) We can rewrite equation (A.1) as \(\frac{\partial F}{\partial t} = w(p)(a + bF - c\sigma(\gamma) \Gamma(t))(1 - F)\). Thus, it immediately follows that:

\[
\frac{\partial^2 F(t\mid t_0, \cdot)}{\partial t \partial t_0} = w(p)[b - bF(t\mid t_0) - a - bF(t\mid t_0) + c\sigma(\gamma) \Gamma(t)] \frac{\partial F(t\mid t_0, \cdot)}{\partial t_0}.
\]
(A.14)

\(^{11}\)See Casella and Berger (2002).
Thus, if $a - b \geq c\sigma(\gamma)$, then, from Lemma 1 we see immediately that $\frac{\partial^2 F(\gamma \mid t_0)}{\partial t_0^2} \geq 0$, with equality only if $t_0 = L(\gamma) > 0$.

(ii) if $\sigma(\gamma) > z_1$, then $b + c\sigma(\gamma) > a$. We distinguish two cases (i) $a > c\sigma(\gamma)$ and (ii) $a \leq c\sigma(\gamma)$. In case (i) we see that $L = 0$. Thus, $\Gamma(L(\gamma)) = 1$. Consequently $b + c\sigma(\gamma)\Gamma(L(\gamma)) - a = b + c\sigma(\gamma) - a > 0$. In case (ii), $L$ satisfies $a = c\sigma(\gamma)\Gamma(L)$. Therefore, in both cases (i) and (ii), it is obvious that $b + c\sigma(\gamma)\Gamma(L(\gamma)) - a > 0$. Thus, if $T$ is small enough, then $b - a - 2bF(t \mid t_0) + c\sigma(\gamma)\Gamma(t) > 0$ for all $t_0 \leq t \leq T$. From (A.14), we see that result holds for such small $T$. □

**Lemma A2** Let $0 < t_1 < t_2$ and $\tau = t_2 - t_1$. If $t_2 > 2\tau$, then $\Gamma(t_1) \geq \frac{e^{(a+\varepsilon)\tau}}{e^{\rho\tau} + e^{c\tau}} \times \Gamma(t_2)$.

**Proof.** Since $\Phi(L(\gamma)) < 0$, there exists $T_0 > L(\gamma)$ such that for $L(\gamma) < T < T_0$ we have $H = T > L(\gamma)$. Define $\Omega(t) = \Phi(t) - p\beta$. Note that $\Omega(t)$ is increasing in $t$ and $\Omega(t) < 0$ for all $t \in [L(\gamma), H]$. From Lemma A1.ii we know there exists $T_1 > L(\gamma)$ such that for $L(\gamma) < T < T_1$ we have $\frac{\partial F(t \mid t_0)}{\partial t_0} < 0$ and decreasing in $t$ over $[L(\gamma), T]$. Pick $L(\gamma) < T < \bar{T} = \min \{T_0, T_1, L(\gamma) + \frac{p}{\Omega(L(\gamma))}\}$. Then $H = T$ and:

$$
\int_{t_0}^{H} e^{-\beta t} \frac{\partial F(t \mid t_0)}{\partial t_0} \Phi(t) dt < \frac{\partial F(T \mid t_0)}{\partial t_0} \int_{t_0}^{H} e^{-\beta t} \Phi(t) dt
$$

$$
= \frac{\partial F(T \mid t_0)}{\partial t_0} \left( \int_{t_0}^{H} e^{-\beta t} (p\beta + \Omega(t)) dt \right)
$$

$$
= \frac{\partial F(T \mid t_0)}{\partial t_0} \left( \int_{t_0}^{H} e^{-\beta t} p\beta dt + \int_{t_0}^{H} e^{-\beta t} \Omega(t) dt \right)
$$

$$
< \frac{\partial F(T \mid t_0)}{\partial t_0} \left[ (e^{-\beta t_0} - e^{-\beta H})p + (H - t_0)\Omega(L(\gamma)) e^{-\beta t_0} \right]
$$

$$
= -p \frac{\partial F(T \mid t_0)}{\partial t_0} e^{-\beta H} + \frac{\partial F(T \mid t_0)}{\partial t_0} e^{-\beta t_0} [p + (H - t_0)\Omega(L(\gamma))]
$$

$$
< -p \frac{\partial F(T \mid t_0)}{\partial t_0} e^{-\beta H},
$$

where the last inequality holds because $(H - t_0) < T - L < \frac{p}{\Omega(L(\gamma))}$, or $p + (H - t_0)\Omega(L(\gamma)) > 0$. In all the above, $\frac{\partial F(t \mid t_0)}{\partial t_0}$ is negative and decreasing, and $\Phi(t)$ and $\Omega(t)$ are negative and increasing for $t \in [t_0, T]$ ($T = H$). Consequently, for any $T < \bar{T}$, it follows from (A.13) that $\frac{\partial F(t \mid t_0)}{\partial t_0} \leq 0$ for any $L(\gamma) < T < \bar{T}$ and all $t_0 \in [L(\gamma), T]$, with equality solely when $t_0 = L(\gamma)$. Consequently, for any $T \in (L(\gamma), \bar{T})$ we have $t_0 = L$.

(ii) If $\sigma(\gamma) < z_1$, then it immediately follows that $a > c\sigma(\gamma)$ and, thus, $L(\gamma) = 0$. Let $1 > \varepsilon > 0$ be given.
Consider $p$ and $C_T$ very small and $T$ very large such that $T > H$ and $H$ is very large such that $\delta \triangleq \frac{\epsilon H}{p + \epsilon} $ is also large enough to satisfy $\delta e^{\delta t} > \left( \frac{e^{(\theta + \xi)\delta}}{e^{\theta \xi} + e^{\delta}} - 1 \right) > 1$. This is possible because $H$ increases as $p$ and $C_T$ get smaller because $\Phi(t)$ is increasing, and, as long as $T$ is very large, we can push $H$ to be very large as well.

We will prove that $\lim_{T \to \infty} \frac{\partial H(t_o)}{\partial t_0} = 0$ for all $t_0 \in [0, H - 2\delta]$.

Let us choose a release time $t_0 \in [0, H - 2\delta]$. Following Lemmas 1 and 1, we know that $\frac{\partial F(t|t_0)}{\partial t_0} < 0$ and is increasing in $t$. Thus $\left| \frac{\partial F(t|t_0)}{\partial t_0} \right|$ is decreasing in $t$ and $\lim_{T \to \infty} m(Y)pe^{-\beta T + \frac{\partial F(T|t_0)}{\partial t_0}} = 0$. Thus, for large values of $T$ the sign of $\frac{\partial H(t_0)}{\partial t_0}$ will be driven by the sign of the second term in (A.13).

Let us decompose and lowerbound the integral in the second term in (A.13) as follows:

$$
\int_{t_0}^{T} e^{-\beta t} \frac{\partial F(t|t_0)}{\partial t_0} \Phi(t) dt = \int_{t_0}^{H} e^{-\beta t} \frac{\partial F(t|t_0)}{\partial t_0} \Phi(t) dt + \int_{H}^{T} e^{-\beta t} \frac{\partial F(t|t_0)}{\partial t_0} \Phi(t) dt
$$

$$
> \int_{H-\delta}^{H-\delta} e^{-\beta t} \frac{\partial F(H-\delta|t_0)}{\partial t_0} \Phi(H-\delta) dt
$$

where the inequalities hold because we consider $t_0 \leq H - 2\delta$, and $\Phi(t) < 0$ for $t < H$ ($\geq H$). Next, we will find separate lower bounds for each of the two terms in (A.15). From Lemma A1 we know that $\frac{\partial F(t|t_0)}{\partial t_0} < 0$ and is increasing in $t$ (i.e., $\left| \frac{\partial F(t|t_0)}{\partial t_0} \right|$ is decreasing in $t$). Moreover, since $\Phi(t) < 0$ and increasing over $[H - 2\delta, H - \delta]$, we have:

$$
\int_{H-\delta}^{H-\delta} e^{-\beta t} \frac{\partial F(H-\delta|t_0)}{\partial t_0} \Phi(H-\delta) dt
$$

$$
= \delta e^{-\beta(H-\delta)} \frac{\partial F(H-\delta|t_0)}{\partial t_0} \Phi(H-\delta)
$$

$$
= \delta e^{-\beta(H-\delta)} \times \left| \frac{\partial F(H-\delta|t_0)}{\partial t_0} \right| \times |\Phi(H-\delta)|
$$

$$
\geq \delta e^{-\beta(H-\delta)} \times \left| \frac{\partial F(H|t_0)}{\partial t_0} \right| \times |\Phi(H-\delta)|. 
$$

(A.16)

Moreover, since $\Phi(H) = 0$, it follows that:

$$
p\beta + \theta_a(Y)\alpha YC_T = \alpha \theta_a(Y)(C_{P,u} - C_{P,f})\bar{B}e^{-\theta H} + C_G B(H), 
$$

(A.17)

and, consequently, we can write:

$$
|\Phi(H - \delta)| = -\Phi(H - \delta) = \alpha \theta_a(Y)(C_{P,u} - C_{P,f})\bar{B}e^{-\theta H} (e^{\delta \bar{T}} - 1) + C_G (B(H - \delta) - B(H)). 
$$

(A.18)

From Lemma A2 we know that $\Gamma(H - \delta) \geq \frac{e^{(\theta + \xi)\delta}}{e^{\theta \xi} + e^{\delta}} \Gamma(H)$ and we know that $B(t) = \bar{B} \Gamma(t)$ for any $t \geq 0$. Moreover, note that $e^{\delta \bar{T}} > \frac{e^{(\theta + \xi)\delta}}{e^{\theta \xi} + e^{\delta}}$ and we chose $H$ high enough such that $\delta$ is high enough to satisfy $\frac{e^{(\theta + \xi)\delta}}{e^{\theta \xi} + e^{\delta}} > 1$. Therefore, using these properties, (A.17), and (A.18), we have:

$$
|\Phi(H - \delta)| \geq \left( \frac{e^{(\theta + \xi)\delta}}{e^{\theta \xi} + e^{\delta}} - 1 \right) \times \left[ \alpha \theta_a(Y)(C_{P,u} - C_{P,f})\bar{B}e^{-\theta H} + C_G B(H) \right]
$$

$$
= \left( \frac{e^{(\theta + \xi)\delta}}{e^{\theta \xi} + e^{\delta}} - 1 \right) \times (p\beta + \theta_a(Y)\alpha YC_T).
$$

(A.19)
From (A.16) and (A.19), we have:

\[
\int_{H-2\delta}^{H-\delta} e^{-\beta t} \frac{\partial F(t|t_0)}{\partial t_0} \Phi(t) dt > \delta e^{-\beta(H-\delta)} \times \left( \frac{e^{(\theta+\xi)\delta}}{e^{\theta \delta} + e^{\delta \xi}} - 1 \right) \times (p_\beta + \theta_u(Y)\alpha Y C_T) \times \left| \frac{\partial F(H|t_0)}{\partial t_0} \right|. \tag{A.20}
\]

Next, we find a lower bound for the second term in (A.15). When \( t \geq H, 0 \leq \Phi(t) \leq p_\beta + \theta_u(Y)\alpha Y C_T \). Therefore, using the monotonicity of and signs of \( \Phi(t) \) and \( \frac{\partial F(t|t_0)}{\partial t_0} \), we have:

\[
\int_H^T e^{-\beta t} \frac{\partial F(t|t_0)}{\partial t_0} \Phi(t) dt > \int_H^{\infty} e^{-\beta t} \frac{\partial F(t|t_0)}{\partial t_0} \Phi(t) dt
\geq \int_H^{\infty} e^{-\beta t} \frac{\partial F(H|t_0)}{\partial t_0} (p_\beta + \theta_u(Y)\alpha Y C_T) dt
= (p_\beta + \theta_u(Y)\alpha Y C_T) \frac{\partial F(H|t_0)}{\partial t_0} \times e^{-\beta H} \beta
= \left| \frac{\partial F(H|t_0)}{\partial t_0} \right| (-p_\beta - \theta_u(Y)\alpha Y C_T) e^{-\beta H} \beta. \tag{A.21}
\]

Given that \( H \) was chosen high enough such that \( \delta \) satisfies \( \delta \beta e^{3\delta} \times \left( \frac{e^{(\theta+\xi)\delta}}{e^{\theta \delta} + e^{\delta \xi}} - 1 \right) > 1 \), using lower bounds (A.20) and (A.21) for the terms in (A.15), we get:

\[
\int_{t_0}^T e^{-\beta t} \frac{\partial F(t|t_0)}{\partial t_0} \Phi(t) dt > 0, \quad \forall t_0 \in [0, H - 2\delta].
\]

Taking \( T \) very large makes the first term in (A.13) vanish given that \( \left| \frac{\partial F(T|t_0)}{\partial t_0} \right| < \left| \frac{\partial F(t_0|t_0)}{\partial t_0} \right| \). Thus, using (A.22), for sufficiently large \( T \) and low enough \( p \) and \( C_T \), for all \( t_0 \in [0, H - 2\delta] \) we have \( \frac{\partial \Pi(t_0)}{\partial t_0} > 0 \). Since \( t_0^* \leq H \), it must be the case that \( t_0^* \in (H - 2\delta, H] \) and thus \( \frac{H-T}{H-L} < \frac{2\delta}{T} = \epsilon \). □

**Proof of Proposition 3.** We use Topkis’ monotonicity theorem.\(^{12}\) For fixed \( p \), we see that profit \( \Pi(t_0) \) is supermodular in \( t_0 \) (trivial since we have only one control variable). Differentiating (A.13) with respect to \( \alpha \) and using the fact that \( \bar{B} = \sigma(\gamma)Y \) we get the following cross-partial derivative:

\[
\frac{\partial^2 \Pi(t_0, \cdot)}{\partial t_0 \partial \alpha} = m(Y)\theta_u(Y)Y \int_{t_0}^{T} \frac{\partial F(t|t_0, \cdot)}{\partial t_0} e^{-\beta t} \left[ C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t} \right] dt.
\]

We know that the integral term is negative due to Lemma 1(i). Using Topkis’ theorem, we see that \( t_0^* \) is non-decreasing (non-increasing) in \( \alpha \) when \( C_T \leq (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T} \) (respectively \( C_T \geq (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-L(\gamma)\theta} \)). □

**Technical formulation of Proposition 4.** Consider the special case when \( m(Y) = m \) and let \( I^Y \) be a functionality region (interval). Then the following hold:

(i) If \( C_T \geq (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta L(\gamma)} \) then

(a) If \( Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < \bar{\kappa} \triangleq \frac{C_0\sigma(\gamma)\Gamma(T)}{\alpha(C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T})} \) for all \( Y \in I^Y \) then \( t_0^* \) is weakly increasing in

Then, it immediately follows that $\partial \Pi$.

(b) If $Y \frac{\partial \theta}{\partial Y} + \theta_a(Y) > \kappa \triangleq \frac{C_G \sigma(\gamma) \Gamma(L(\gamma))}{\alpha (C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T} - C_T)}$ or all $Y \in I^Y$ then $t_0^*$ is weakly decreasing in $Y$ over $I^Y$.

(ii) If $C_T < (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T}$ then:

(a) If $Y \frac{\partial \theta_a}{\partial Y} + \theta_a(Y) < \mu \triangleq -\frac{C_G \sigma(\gamma) \Gamma(L(\gamma))}{\alpha (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T} - C_T}$ or all $Y \in I^Y$ then $t_0^*$ is weakly decreasing in $Y$ over $I^Y$.

(b) If $Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \geq 0$ for all $Y \in I^Y$ then $t_0^*$ is weakly increasing in $Y$ over $I^Y$.

**Proof.** Profit is supermodular in $t_0$ (unique control). When $m(Y) = m$, using (A.13) and the fact that $F$ is independent of $Y$ (see the derivation of path $N(t)$ at the beginning of the Appendix) we obtain the following cross-partial of profit with respect to $t_0$ and $Y$:

$$
\frac{\partial^2 \Pi(t_0; \gamma)}{\partial t_0 \partial Y} = m \int_{t_0}^T \frac{\partial F}{\partial t_0}(t|t_0, \gamma)e^{-\theta t} \left\{ \alpha \left( Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \right) (C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t}) - C_G \sigma(\gamma)\Gamma(t) \right\} dt. \quad (A.22)
$$

(i) When $C_T > (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta L(\gamma)}$, then $C_T > (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t}$ for all $t \geq t_0$.

(a) If for some $Y \in I^Y$ we have $Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \leq 0$, then it immediately follows that $C_G \sigma(\gamma)\Gamma(t) \geq \alpha \left( Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \right) (C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t})$ for all $t \geq t_0$. Otherwise, if $Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) > 0$, given that $\Gamma(t)$ is decreasing, we have:

$$
C_G \sigma(\gamma)\Gamma(t) \geq C_G \sigma(\gamma)\Gamma(T) > \alpha \left( Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \right) (C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T}) > \alpha \left( Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \right) (C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t}).
$$

Then, in both cases, it immediately follows that $\frac{\partial^2 \Pi(t_0; \gamma)}{\partial t_0 \partial Y} > 0$ for all $Y \in I^Y$. By applying Topkis’ Theorem we see that $t_0^*$ is non-decreasing in $Y$ over $I^Y$.

(b) If $Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) > \kappa$ then, for any $t \geq L(\gamma)$ it follows that $\alpha \left( Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \right) [C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t}] > C_G \sigma(\gamma)\Gamma(t)$. We obtain the desired result by applying Topkis’ Theorem.

(ii) When $C_T < (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T}$, then $C_T < (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t}$ for all $t \geq t_0$.

(a) For any $Y \in I^Y$ and $t \in [L(\gamma), T]$ we have:

$$
\alpha \left( Y \frac{\partial \theta_a(Y)}{\partial Y} + \theta_a(Y) \right) (C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t}) > \alpha(-\bar{\mu})[(C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T} - C_T]
$$

$$
= C_G \sigma(\gamma)\Gamma(L(\gamma))
$$

$$
\geq C_G \sigma(\gamma)\Gamma(t).
$$

Then, it immediately follows that $\frac{\partial^2 \Pi(t_0; \gamma)}{\partial t_0 \partial Y} < 0$ for all $Y \in I^Y$. By applying Topkis’ Theorem we see that $t_0^*$ is non-increasing in $Y$ over $I^Y$.

(b) It immediately follows that $\frac{\partial^2 \Pi(t_0; \gamma)}{\partial t_0 \partial Y} > 0$ for all $Y \in I^Y$. We obtain the desired result by applying Topkis’ Theorem. \(\square\)
Proof of Proposition 5. Under our simplified framework, as the result of introducing a Beta test group of size \( N_B \), for any \( t \in [0, t_0) \) the firm adjusts its internal testing efforts to \( \theta_f(t) = \theta - \alpha \theta_u(Y)N_B \). Thus, testing cost \( \tilde{C}_T \) and bug processing cost \( \tilde{C}_P \) change as follows:

\[
\tilde{C}_T = \int_0^T C_T \times \theta_f(t) \times Y \times e^{-\beta t} \, dt
\]

\[
= C_T Y \left( \int_0^T (\theta - \alpha \theta_u(Y)N_B)e^{-\beta t} \, dt + \int_{t_0}^T (\theta - \alpha \theta_u(Y))N(t)e^{-\beta t} \, dt \right)
\]

\[
= C_T Y \left( \int_0^T \theta e^{-\beta t} \, dt - \int_{t_0}^T \alpha \theta_u(Y)N(t)e^{-\beta t} \, dt - \int_{t_0}^T \theta \theta_u(Y)N(t)e^{-\beta t} \, dt \right)
\]

\[
= \frac{C_T Y}{\beta} \left[ \theta \left( 1 - e^{-\beta T} \right) - \alpha \theta_u(Y)N_B \left( 1 - e^{-\beta t_0} \right) \right] - C_T \theta_u(Y) \alpha Y \int_{t_0}^T N(t)e^{-\beta t} \, dt,
\]

(A.23)

\[
\tilde{C}_P = \int_0^T \left( \frac{C_{P,f} \times \theta_f(t)}{\theta} + \frac{C_{P,u} \times \theta - \theta_f(t)}{\theta} \right) \times B\theta e^{-\beta t} \times e^{-\beta t} \, dt
\]

\[
= \int_0^T \left( C_{P,f} \times \frac{\theta - \alpha \theta_u(Y)N_B}{\theta} + \frac{C_{P,u} \times \alpha \theta_u(Y)N_B}{\theta} \right) \times B\theta e^{-\beta t} \times e^{-\beta t} \, dt
\]

\[
+ \int_{t_0}^T \left( C_{P,f} \times \frac{\theta - \alpha \theta_u(Y)N(t)}{\theta} + C_{P,u} \times \frac{\alpha \theta_u(Y)N(t)}{\theta} \right) \times B\theta e^{-\beta t} \times e^{-\beta t} \, dt
\]

\[
= \int_0^T C_{P,f} B\theta e^{-(\beta+\theta)'t} \, dt + \int_{t_0}^T (C_{P,u} - C_{P,f}) \alpha \theta_u(Y)N_B \tilde{B} e^{-(\beta+\theta)'t} \, dt
\]

\[
+ \int_{t_0}^T (C_{P,u} - C_{P,f}) \alpha \theta_u(Y) \tilde{B} e^{-(\beta+\theta)'t} \, dt
\]

\[
= \frac{C_{P,f} B\theta (1 - e^{-(\beta+\theta)'T})}{\theta + \beta} + (C_{P,u} - C_{P,f}) \alpha \theta_u(Y) \tilde{B} N_B \left( 1 - e^{-(\beta+\theta)'t_0} \right) \frac{1}{\theta + \beta}
\]

\[
+ (C_{P,u} - C_{P,f}) \alpha \theta_u(Y) \tilde{B} \int_{t_0}^T N(t)e^{-(\beta+\theta)'t} \, dt.
\]

(A.24)

Computing the cross-partial derivative of profit with respect to \( t_0 \) and \( N_B \), we obtain:

\[
\frac{\partial \Pi(t_0, N_B)}{\partial t_0 \partial N_B} = Y \alpha \theta_u(Y)e^{-\beta t_0} \left[ C_T - (C_{P,u} - C_{P,f}) \alpha (\gamma)e^{-\beta t_0} \right].
\]

(A.25)

From Topkis’ theorem, it follows that (i) when \( \sigma(\gamma) > \frac{C_T \sigma(\gamma) e^{\beta T}}{C_{P,u} - C_{P,f}} \) then \( t_0^* \) is weakly decreasing in \( N_B \), and (ii) when \( \sigma(\gamma) < \frac{C_T \sigma(\gamma) e^{\beta L(\gamma)}}{C_{P,u} - C_{P,f}} \) then \( t_0^* \) is weakly increasing in \( N_B \). \( \square \)

Lemma A3 Consider the multivariate optimization in (18). Define \( \hat{p} \) as the solution to the equation \( p \bar{w}'(p) + w(p) = 0 \) and consider \( z_1 \) as defined in Proposition 2. The following hold true:

(i) If \( \sigma(\gamma) < z_1 \) and \( C_G \Gamma(T) > \alpha \theta_u(Y) \left[ \frac{C_T}{\sigma(\gamma)} - (C_{P,u} - C_{P,f}) e^{-\theta T} \right] \), then the firm prices at \( p^* > \hat{p} \).

Moreover, for any release time \( t_0 < T \), under the above conditions, we have \( \frac{\partial \pi(t_0, p)}{\partial p} > 0 \) when \( p \leq \hat{p} \).

(ii) If \( z_1 < \sigma(\gamma) \), \( C_G \Gamma(L(\gamma)) \leq \alpha \theta_u(Y) \left[ \frac{C_T}{\sigma(\gamma)} - (C_{P,u} - C_{P,f}) e^{-\theta L(\gamma)} \right] \), and short product horizon \( T \), we

A.8
have \( p^* \leq \tilde{p} \). Moreover, for any release time \( t_0 < T \), under the above conditions, we have \( \frac{\partial \pi(t_0,p)}{\partial p} < 0 \) when \( p \geq \tilde{p} \).

**Proof.** (i) For the beginning, let us fix \( t_0 \geq L(\gamma) \). We can write revenue \( \int_{t_0}^{T} pN'(t|p,\cdot)e^{-\beta t} dt = pm(Y)F(T|p,\cdot)e^{-\beta T} + m(Y)p\beta \int_{t_0}^{T} F(t|p,\cdot)e^{-\beta t} dt \). We can use this to regroup the revenue terms in (A.12).

Next, differentiating (A.12) with respect to price, we obtain:

\[
\frac{\partial \Pi(t_0,p)}{\partial p} = m(Y) \int_{t_0}^{T} \left( f(t|p,\cdot) + p \frac{\partial f}{\partial p}(t|p,\cdot) \right) e^{-\beta t} dt
\]

\[
+ m(Y) \int_{t_0}^{T} \frac{\partial F(t|p,\cdot)}{\partial p} e^{-\beta t} \left( \theta_u(Y)\alpha Y [C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\beta t}] - C_G B(t) \right) dt. \tag{A.26}
\]

First, we inspect the sign of \( f(t|p,\cdot) + p \frac{\partial f}{\partial p}(t|p,\cdot) \). Moving \( 1 - F(t|\cdot) \) in (A.1) on the RHS and differentiating with respect to \( p \) we obtain:

\[
\frac{\partial f}{\partial p}(t|p,\cdot) = \frac{\partial F}{\partial p}(t|p,\cdot) \left( b - a - 2b F(t|p,\cdot) + c \frac{B(t)}{Y} \right) w(p) + f(t|p,\cdot) \frac{w'(p)}{w(p)}. \tag{A.27}
\]

Therefore, we have:

\[
f(t|p,\cdot) + p \frac{\partial f}{\partial p}(t|p,\cdot) = p \frac{\partial F}{\partial p}(t|p,\cdot) \left( b - a - 2b F(t|p,\cdot) + c \frac{B(t)}{Y} \right) w(p) + f(t|p,\cdot) \frac{w'(p)}{w(p)} \left( p + \frac{w(p)}{w'(p)} \right). \tag{A.28}
\]

Since \( 2 > \frac{w'(p)}{w(p)} \), we see that function \( g(p) = p + \frac{w(p)}{w'(p)} \) is strictly increasing and we know that \( g(\tilde{p}) = 0 \). Therefore, when \( p < \tilde{p} \), we have \( g(p) < 0 \). Moreover, when \( \sigma(\gamma) \leq z_1 \), since \( B(t) = \sigma(\gamma)YT(t) \), it can be easily seen that \( b - a + c \frac{B(t)}{Y} < 0 \) for all \( t > t_0 \) because \( \Gamma(t) \) is decreasing in \( t \). We know from Lemma 1 that \( \frac{\partial F}{\partial p}(t|p,\cdot) \leq 0 \) for \( t \geq t_0 \) and from the model hypothesis we know that \( w'(p) < 0 \). Consequently, we have:

\[
f(t|p,\cdot) + p \frac{\partial f}{\partial p}(t|p,\cdot) > 0, \quad \forall \ p \leq \tilde{p}. \tag{A.29}
\]

Under both cases (i.a) and (i.b) it can be shown that \( C_G \Gamma(T) > \alpha \theta_u(Y) \left[ C_T - (C_{P,u} - C_{P,f})e^{-\beta T} \right] \), and it immediately follows that \( C_G B(t) > \alpha \theta_u(Y)Y[C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\beta t}] \) for all \( t \in [t_0, T] \) since \( B(t) = B\Gamma(t) \) and \( e^{-\beta t} \) are both decreasing in \( t \). The argument is easy to see by moving \( \alpha \theta_u(Y)Y(C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\beta t} \) on LHS. Given that \( \frac{\partial F}{\partial p}(t|p,\cdot) \leq 0 \) for \( t \geq t_0 \), from (A.26) we immediately see that

\[
\frac{\partial \Pi(t_0,p)}{\partial p} > 0, \quad \forall \ p \leq \tilde{p}. \tag{A.30}
\]

The above result holds for any \( t_0 \). Replacing \( t_0 \) with \( t_0^* \) we obtain that \( p^* \geq \tilde{p} \).

(ii) In this case, it can be shown that for any \( t \geq L(\gamma) \), we have \( \theta_u(Y)\alpha C_T Y - \theta_u(Y)\alpha(C_{P,u} - C_{P,f})Y\sigma(\gamma)e^{-\beta t} - C_G B(t) \geq 0 \). We immediately see from Proposition 1 that \( t_0^* = L(\gamma) = H(Y,\gamma,p) \) regardless of any value \( p \geq 0 \) (and, implicitly \( p^* \)). Following a similar argument as in the proof of Lemma A1.ii, if \( \sigma(\gamma) > z_1 \) and \( T \) is small enough, then \( b - a - 2bF(t|t_0^*) + c \frac{B(t)}{Y} > 0 \) for all \( L(\gamma) \leq t \leq T \).
If \( p > \tilde{p} \), following a similar argument as in the proof of part (i) we obtain \( f(t)p + p\frac{\partial f}{\partial p}(t)p < 0 \). Consequently, from (A.26), we see that:

\[
\frac{\partial \Pi(\tilde{t}_0, p)}{\partial p} < 0, \quad \forall \ p > \tilde{p}.
\] (A.31)

Therefore, it must be the case that \( p^* \leq \tilde{p} \). \( \square \)

**Technical formulation of Proposition 6.** The following hold true:

(i) If \( \sigma(\gamma) < \frac{C_G}{\alpha\gamma} + \alpha\theta_u(\gamma) \left[ \frac{C_T}{\gamma} - (C_{P,u} - C_{P,f})e^{-\theta L(\gamma)} \right] \), and \( C_G > \alpha\theta_u(\gamma) \left[ \frac{C_T}{\gamma} - (C_{P,u} - C_{P,f})e^{-\theta L(\gamma)} \right], \) then \( \tilde{t}_0 = L(\gamma) \).

(ii) If \( \frac{C_G}{\alpha\gamma} + \alpha\theta_u(\gamma)(C_{P,u} - C_{P,f})e^{-\theta L(\gamma)} < \sigma(\gamma) < z_1 \) and \( \tilde{p}\beta > Y[\left( C_G + \alpha\theta_u(\gamma)(C_{P,u} - C_{P,f})\right] \sigma(\gamma) - \alpha\theta_u(\gamma)C_T \), then \( \tilde{t}_0 = L(\gamma) = 0 \).

(iii) If \( C_G < \alpha\theta_u(\gamma) \left[ \frac{C_T}{\gamma} - (C_{P,u} - C_{P,f}) \right] \), then for any \( \gamma \), \( \tilde{t}_0 = L(\gamma) \).

**Proof.** (i) and (iii). Since \( \Gamma(L(\gamma)) \leq 1 \), \( \sigma(\gamma) \leq \tilde{\sigma} \), and \( e^{-\theta L(\gamma)} < 1 \), we can immediately see that under both cases we have \( C_G \Gamma(L(\gamma)) < \alpha\theta_u(\gamma) \left[ \frac{C_T}{\gamma} - (C_{P,u} - C_{P,f})e^{-\theta L(\gamma)} \right] \). In this case, for any \( t \geq t_0 \), we have \( \theta_u(\gamma)\alpha C_T Y \geq \theta_u(\gamma)\alpha(C_{P,u} - C_{P,f})Y(\gamma)e^{-\theta t} + C_G B(t) \). We immediately see from Proposition 1 that \( \tilde{t}_0 = L(\gamma) = H(Y, \gamma, p) \) regardless of any value \( p \geq 0 \) (and, implicitly \( p^* \)).

(ii) As per Lemma A3, we have \( p^* \geq \tilde{p} \). Since \( \tilde{p}\beta > Y[\left( C_G + \alpha\theta_u(\gamma)(C_{P,u} - C_{P,f})\right] \sigma(\gamma) - \alpha\theta_u(\gamma)C_T \), then, for any \( t \geq t_0 \), we have \( \tilde{p}\beta + \theta_u(\gamma)\alpha Y \left[ C_T - (C_{P,u} - C_{P,f})\right] \sigma(\gamma)e^{-\theta t} - C_G B(t) > 0 \). Given \( p^* > \tilde{p} \), from Proposition 1 we immediately see that \( \tilde{t}_0 = L(\gamma) = H(Y, \gamma, p^*) \). Moreover, if \( z_1 > \sigma(\gamma) \), then \( a > b + \sigma(\gamma) > c\sigma(\gamma) \) and it immediately follows that \( L(\gamma) = 0 \). \( \square \)

**Proof of Proposition 7.** (i, ii, iii) From Proposition 6, we see that in both there cases we have \( \tilde{t}_0 = L(\gamma) \). Thus, the profit maximization can be written in terms of a single variable, price, and, thus, the objective function is by default supermodular in \( p \) and we can apply Topkis’ Theorem. The monotonicity in alpha follows via a two-sided argument similar to that in the proof of Proposition 3. The monotonicity with respect to costs follows in a similar way but there is only one case to be analyzed. \( \square \)

**Technical formulation of Proposition 8.** Consider the special case when \( m(Y) = m \) (constant). Let \( I^Y \) be a functionality region (interval) and \( \bar{\kappa}, \underline{\kappa}, \bar{\mu}, \underline{\mu} \) as defined in the technical formulation of Proposition (4). Then:

(i) Under the conditions in parts (i), (ii), and (iii) of Proposition 6, if \( \sigma(\gamma) < \frac{C_Te^{\theta L(\gamma)}}{C_{P,u} - C_{P,f}} \) then:

(a) if \( Y\frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < \bar{\kappa} \) for all \( Y \in I^Y \) then \( p^* \) is weakly increasing in \( Y \).

(b) if \( Y\frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) > \underline{\kappa} \) for all \( Y \in I^Y \) then \( p^* \) is weakly decreasing in \( Y \).

(ii) Under the conditions in part (ii) in Proposition 6, if \( \frac{C_Te^{\theta T}}{C_{P,u} - C_{P,f}} < \sigma(\gamma) < z_1 \) then:

(a) if \( Y\frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) < \bar{\mu} \) for all \( Y \in I^Y \) then \( p^* \) is weakly decreasing in \( Y \).
(b) if \( Y \frac{\partial \theta_u(Y)}{\partial Y} + \theta_u(Y) \geq 0 \) for all \( Y \in I^Y \) then \( p^* \) is weakly increasing in \( Y \).

**Proof.** Under all cases, from Proposition 6, we know that \( t_0^* = L(\gamma) \). Thus, the profit maximization can be written in terms of a single variable, price, and, thus, the objective function is by default supermodular in \( p \) and we can apply Topkis’ Theorem.

(i) Proof is similar to that of Proposition 4, part (i).

(ii) Proof is similar to that of Proposition 4, part (ii). □

**Proof of Proposition 9.** Suppose we are considering two different release times \( L(\gamma) \leq t_{01} < t_{02} \leq T(Y) \). Then, we define,

\[
\bar{p}(t|t_{01}) = \begin{cases} 
\infty, & \text{for } t \in [t_{01}, t_{02}), \\
p^*(t|t_{02}), & \text{for } t \geq t_{02}.
\end{cases}
\]

It is easy to see that, by releasing at time \( t_{01} \) and using \( \bar{p}(t|t_{01}) \), we obtain the same adoption curve and profit as in the case of an optimal \( t_{02} \) release. The formal argument involves assuming a very large price over \( [t_{01}, t_{02}) \) and taking it to \( \infty \) in the limit. Because of condition \( \lim_{p \to \infty}pw(p) = 0 \) adoption and revenues between \( t_{01} \) and \( t_{02} \) are negligible. Therefore, the optimal pricing policy for a \( t_{01} \) release yields at least the same discounted profit as an optimal pricing policy for a \( t_{02} \) release. Thus, it is always better to release earlier and shape demand solely through price. □

**Lemma A4** The terminal price of the software satisfies \( p^*(T) = \tilde{p} > 0 \).

**Proof.** Since \( t_0^* = L(\gamma) \) (as per Proposition 9), \( p(t) \) is the unique control. After eliminating from the cost functions (A.8)-(A.11) the terms that are not affected by \( p(\cdot) \), the simplified profit maximization problem can be formulated as:

\[
\max_{p(\cdot)} \int_{L(\gamma)}^{T} e^{-\beta t} \left\{ p(t)N'(t) + \left[ \theta_u(Y)\alpha Y \left( C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t} \right) - C_G B(t) \right] N(t) \} \right\} dt
\]

s.t.

\[
N'(t) = \left( m(Y) - N(t) \right) \left( a + \frac{b}{m(Y)} N(t) - \frac{cB(t)}{Y} \right) w(p(t)),
\]

\[
N(L(\gamma)) = 0.
\]

We drop parameter \( t \) for brevity, unless its presence contributes to the clarity of the presentation. We use the maximum principle (e.g., see Sethi and Thompson 2000). The current value Hamiltonian is:

\[
H_C = pN' + \left\{ \theta_u(Y)\alpha Y \left[ C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t} \right] - C_G B \right\} N + \lambda N',
\]  \hspace{1cm} (A.32)
with adjoint equation and transversality condition:

\[
\lambda' = \beta \lambda - \frac{\partial H_C}{\partial N} = \beta \lambda - (p + \lambda)w(p) \left( b - a + \frac{cB}{Y} - 2N \frac{b}{m(Y)} \right) \\
+ CGB - \theta_u(Y)\alpha Y \left[ C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta t} \right]. \tag{A.33}
\]

\[
\lambda(T) = 0.
\]

We know that the optimal price \( p^*(t) \) maximizes the Hamiltonian pointwise. The conditions imposed on \( w \) ensure the existence of an interior solution. We obtain the following pointwise FOC:

\[
0 = \frac{\partial H_C}{\partial p} \bigg|_{p=p^*} = (p^* + \lambda)w'(p^*) + w(p^*) \left( m(Y) - N \right) \left( a + \frac{b}{m(Y)}N - \frac{cB}{Y} \right).
\]

However, since \( a \geq \frac{cB(L(\gamma))}{\gamma} \), we obtain that \( (m(Y) - N) \times \left( a + \frac{b}{m(Y)}N - \frac{cB(t)}{Y} \right) > 0 \) for all \( t > L(\gamma) \). This yields the following optimal path condition:

\[
(p^*(t) + \lambda(t))w'(p^*(t)) + w(p^*(t)) = 0, \quad \forall \ t \geq L(\gamma), \tag{A.34}
\]

where equality at \( L(\gamma) \) is obtained by continuity. Therefore we have:

\[
p^* + \lambda = -\frac{w(p^*)}{w'(p^*)} > 0. \tag{A.35}
\]

The condition \( \frac{w''}{(w')^2} < 2 \), along with (A.35) ensure that \( \frac{\partial^2 H_C}{\partial p^2} < 0 \). Therefore, the second order condition for optimality is satisfied. Since \( \lambda(T) = 0 \), consequently, from (A.34) and (A.35), we have \( p^*(T) > 0 \) and \( p^*(T)w'(p^*(T)) + w(p^*(T)) = 0 \). Last, we discuss the uniqueness of the solution. Since \( \frac{w''}{(w')^2} < 2 \), we see that the function \( x + \frac{w(x)}{w'(x)} \) is strictly increasing. Therefore, the equation \( x = -\frac{w(x)}{w'(x)} \) has a unique positive solution. \( \square \)

**Lemma A5** Over the interval \([L(\gamma), T]\) we have \( \text{sign} (p^*) = -\text{sign}(\lambda') \).

**Proof.** Differentiating (A.34) with respect to time, we obtain:

\[
2p^*w'(p^*) + (p^* + \lambda)w''(p^*)p^* + \lambda'w'(p^*) = 0. \tag{A.36}
\]

Using (A.34) and the fact that \( w'(\tilde{p}^*_u) < 0 \), we obtain:

\[
p^* \left( 2 - \frac{w(p^*)w''(p^*)}{(w'(p^*))^2} \right) = -\lambda'. \tag{A.37}
\]

Given that \( 2 > \frac{w''}{(w')^2} \), the result follows immediately. \( \square \)

A.12
Lemma A6 For all $t \in [L(\gamma), T]$, the shadow price $\lambda$ satisfies the following equation:

$$\lambda(t) = -\int_t^T h(r)e^{-\beta r}dr,$$

where:

$$h(t) \triangleq \frac{w^2(p^*(t))}{w'(p^*(t))} \left( b - a + c \frac{B(t)}{Y} - \frac{2bN(t)}{m(Y)} \right) + CGB(t) - \theta_u(Y)\sigma(Y) \left[ CT - (CP_u - CP_f)\sigma(\gamma) e^{-\theta T} \right].$$

Proof. Using (A.35), equation (A.33) can be rewritten as:

$$\lambda' = \beta \lambda + h.$$  (A.40)

Using the integration factor $e^{-\beta t}$, we obtain:

$$\lambda(t)e^{-\beta t} = \int_{L(\gamma)}^t h(r)e^{-\beta r}dr + K,$$

where $K$ is a constant. Using the terminal condition $\lambda(T) = 0$, we obtain $K = -\int_{L(\gamma)}^T h(r)e^{-\beta r}dr$, and the result follows. □

Proof of Proposition 10. (i) Since $\lambda(T) = 0$, from (A.33) we obtain: $\lambda'(T) = -h(T)$. When $\sigma(\gamma) \leq z_2(Y)$, we have $b - a + cB(T) \leq 0$. Since $CG\Gamma(T) > \alpha\theta_u(Y) \left[ \frac{C_P}{\sigma(\gamma)} - (CP_u - CP_f) e^{-\theta T} \right]$, it immediately follows that $h(T) > 0$. Since $\lambda(T) = 0$, from (A.40) we see that $\lambda'(T) = h(T) > 0$. From Lemma A5 it follows that $p''(T) < 0$. Therefore, there exists $L(\gamma) \leq t_1 < T$ such that optimal price $p^*(t)$ is decreasing (and positive because $p^*(T) > 0$) over the interval $[t_1, T]$.

(ii) When $\sigma(\gamma) \leq z_1$, we have $b - a + c\sigma(\gamma) \leq 0$. Thus, $a > c\sigma(\gamma)$ and, implicitly, $L(\gamma) = 0$.

(ii.a) Since $\Gamma(t) \leq 1$ for all $t \geq 0$, it immediately follows that $b - a + c\frac{B(t)}{Y} \leq 0$. When $CG\Gamma(T) > \alpha\theta_u(Y) \left[ \frac{C_P}{\sigma(\gamma)} - (CP_u - CP_f) e^{-\theta T} \right]$, it can be easily shown that $CGB(t) - \theta_u(Y)\sigma(Y) \left[ CT - (CP_u - CP_f)\sigma(\gamma) e^{-\theta T} \right] \geq 0$ for all $t \in [0, T]$ since $\Gamma(t)$ and $e^{-\theta t}$ are both decreasing in $t$. Thus, $h(t) > 0$ for all $t \in [0, T]$. Using (A.38), we can see that $\lambda(t) \leq 0$ for all $t \in [0, T]$. From (A.35), we know that $p^* + \lambda > 0$. Therefore, for all $t \in [0, T]$ we have $p^*(t) > 0$.

(ii.b) Under no discounting ($\beta = 0$), using (A.40) and the proof in part (ii.a), it immediately follows that $\lambda'(t) = h(t) > 0$, $\forall \ t \in [0, T]$. Thus, using Lemma A5, we see that the optimal price $p^*(t)$ is decreasing on the entire interval $[0, T]$. □

Proof of Proposition 11. (i.a) We are going to prove this result by contradiction. Suppose the contrary, i.e., that $p^*$ is decreasing over the entire product life. From Lemma A5, it must be the case that $\lambda$ is weakly
increasing over the entire product life horizon, i.e. \( \lambda'(t) \geq 0 \) for all \( t \in [L(\gamma), T] \). Since \( \lambda(T) = 0 \), it must be the case that:

\[
\lambda(t) \leq 0, \text{ for all } t \in [L(\gamma), T]. \tag{A.41}
\]

When \( z_1 < \sigma(\gamma) \), we have \( b - a + c\frac{B}{\bar{Y}} > 0 \). If \( L(\gamma) = 0 \), then \( B(0) = \bar{B} \). If \( L(\gamma) > 0 \), then \( a = c\frac{B(L(\gamma))}{\gamma} \). Therefore, regardless of the value of \( L(\gamma) \), we have \( b - a + c\frac{B(L(\gamma))}{\gamma} > 0 \). Moreover, when \( C_G < \theta_u(Y)\alpha\frac{C_p}{\sigma} - (C_{P,u} - C_{P,f}) \), it can be shown that for any \( t \geq 0 \), we have \( \theta_u(Y)\alpha C_T Y - \theta_u(Y)\alpha(C_{P,u} - C_{P,f})Y \sigma(\gamma)e^{-\theta t} - C_G B(t) \geq 0 \). From (A.39), we immediately see that \( h(L(\gamma)) < 0 \). Therefore, from (A.40), applying (A.41), we see that it must be the case that:

\[
\lambda'(L(\gamma)) < 0.
\]

This is a contradiction. Therefore, \( p^* \) must be increasing at least on some regions of the product life cycle.

(i.b) When \( \beta = 0 \), from (A.40) we see that the sign of \( \lambda' \) depends solely on the sign of \( h \). From part (a) we saw that \( h(L(\gamma)) < 0 \). Consequently \( \lambda'(L(\gamma)) < 0 \), and, thus, \( \lambda \) is strictly decreasing immediately after market release. Consequently, from (A5), \( p^* \) is strictly increasing immediately after release.

(ii) Scenario 1: \( C_T \) high.

We know from Lemma A4 that \( p^*(T) > 0 \). Note that the function \( \zeta(p) \triangleq \frac{w^2(p)}{w'(p)} \) is negative and strictly increasing in \( p \) because \( w' < 0 \) and we assumed that \( 2 > \frac{w''}{w'} \). Moreover \( \zeta(p) < 0 \). Since \( p^*(T) > 0 \), we see that \( |\zeta(p^*(T))| < \left| \frac{w^2(0)}{w'(0)} \right| \). Moreover, since \( a \geq c\frac{B(L(\gamma))}{\gamma} \) and \( 0 \leq N(t) \leq m(Y) \), for all \( t \in [L(\gamma), T] \), we have

\[
a + b > b \geq b - a + c\frac{B(t)}{Y} - 2b\frac{N(t)}{m(Y)} \geq b - a - 2b = a - b,
\]

which can be rewritten as:

\[
\left| b - a + c\frac{B(t)}{Y} - 2b\frac{N(t)}{m(Y)} \right| \leq a + b.
\]

Define:

\[
Q \triangleq \left| \frac{w^2(0)}{w'(0)} \right| \times (a + b). \tag{A.42}
\]

Consider \( C_T \) high enough such that:

\[
Q < -C_G B(T) + \alpha \theta_u(Y) Y \left[ C_T - (C_{P,u} - C_{P,f})\sigma(\gamma)e^{-\theta T} \right]. \tag{A.43}
\]

From (A.39), it immediately follows that:

\[
h(T) < Q - Q = 0. \tag{A.44}
\]
Using the fact that \( \lambda(T) = 0 \) in (A.40), we obtain:

\[
\lambda'(T) = h(T) < 0. \tag{A.45}
\]

Therefore, there must exist \( t_2 < T \) such that \( \lambda \) is strictly decreasing on the interval \([t_2, T]\). Consequently, from (A5), \( p^* \) is be strictly increasing in the interval \([t_2, T]\).

**Scenario 2:** \( C_T \) low (but still \( C_G < \alpha \theta_u(Y) \left[ \frac{C_T}{\sigma} - (C_{P,u} - C_{P,f}) \right] \)), \( T \) high, and \( a > b \).

Consider \( \zeta(p) \) as defined in the above discussion for Scenario 1. Note that \( |\zeta(p^*(T))| > 0 \) and \( p^*(T) \) is independent of \( T \), as shown in Lemma A4. Consider \( T \) large enough (as a function evaluated at \( Y \)) and \( \rho > 0 \) small enough such that \( \beta = \frac{B(T)}{\zeta(p^*(T))} < \frac{\rho}{\zeta(p^*(T))} < 0 \). We can find \( T \) satisfying the above inequality because we consider a case where \( a > b \) and \( B(t) \) is decreasing in \( t \) with \( \lim_{t \to \infty} B(t) = 0 \). Thus, we have:

\[
\frac{w^2(p^*(T))}{w'(p^*(T))} \times \left( b - a + \frac{B(T)}{Y} - 2bN(T)\frac{N(T)}{m(Y)} \right) > \rho. \tag{A.46}
\]

Pick \( C_T < \frac{\alpha \theta_u(Y)}{Y \theta_u(Y)} \), which implies that \( C_G \) and \( C_{P,u} - C_{P,f} \) are also small enough given that we impose \( C_G < \alpha \theta_u(Y) \left[ \frac{C_T}{\sigma} - (C_{P,u} - C_{P,f}) \right] \). Then, using the argument in the proof of part (i.a), we have:

\[
0 > C_G B(T) - \alpha \theta_u(Y) Y \left[ C_T - (C_{P,u} - C_{P,f}) \sigma(\gamma) e^{-\theta T} \right] > -\rho. \tag{A.47}
\]

Using (A.39), (A.46), and (A.47), we see that:

\[
h(T) > 0.
\]

Using the fact that \( \lambda(T) = 0 \) in (A.40), we obtain \( \lambda'(T) = h(T) > 0 \). Therefore, there must exist \( t_3 < T \) such that \( \lambda \) is strictly increasing on the interval \([t_3, T]\). Consequently, from (A5), \( p^* \) is be strictly decreasing in the interval \([t_3, T]\). □

**Proposition A1** In the special case where \( \beta, C_G, \) and \( \alpha \) are zero, the optimal price path \( p^*(t) \) is:

(i) Unimodal if software process maturity is intermediate, i.e. \( z_1 < \sigma(\gamma) < z_2 \);

(ii) Either increasing or unimodal if software process maturity is low, i.e. \( z_2 \leq \sigma(\gamma) \).

**Proof.** Under no discounting (\( \beta = 0 \)), the first term (\( \beta \lambda \)) vanishes from the expression of \( \lambda' \) in equation (A.33), and we have:

\[
\lambda'(t) = -(p^*(t) + \lambda(t))w(p^*(t)) \left( b - a + c\frac{B(t)}{Y} - 2N(t)\frac{b}{m(Y)} \right). \tag{A.48}
\]

Since \( (p^*(t) + \lambda(t))w(p^*(t)) > 0 \), the sign of \( -\lambda'(t) \), or, according to Lemma A5, the sign of \( p^{*'}(t) \), is given by the sign of \( b - a + c\frac{B(t)}{Y} - 2N(t)\frac{b}{m(Y)} \). All the derivations in the proofs of Propositions 10 and 11 are still valid.
(i) When $\sigma(\gamma) < z_2$, it follows that $b - a + c\frac{B(T)}{Y} < 0$. It immediately follows that $\lambda'(T(Y)) > 0$. When $z_1 < \sigma(\gamma)$, then $a - b < c\sigma(\gamma) = \frac{cB}{Y}$. If $L(\gamma) = 0$, then $B(L(\gamma)) = \bar{B}$. Otherwise, if $L(\gamma) > 0$, then $a = c\frac{B(L(\gamma))}{Y}$. Therefore, regardless of the value of $L(\gamma)$, we have $b - a + c\frac{B(L(\gamma))}{Y} > 0$. Since $N(L(\gamma)) = 0$, we have:

$$b - a + c\frac{B(L(\gamma))}{Y} - 2N(L(\gamma))\frac{b}{m(Y)} > 0$$

Thus, since $b - a + c\frac{B(t)}{Y} - 2N(t)\frac{b}{m(Y)}$ is decreasing in $t$, $\lambda'$ changes sign exactly once in the interval $[L(\gamma), T]$. Let $\tilde{t}$ be the point where $\lambda'(\tilde{t}) = 0$. We see that $\lambda'(t) < 0$ in $[L(\gamma), t_1)$ and $\lambda'(t) > 0$ on $(t_1, T]$. Since the sign of $p^{*\prime}$ is the opposite of the sign of $\lambda'$, the proof of part (i) is complete.

(ii) In this case, we see that note that $b - a + c\frac{B(T)}{Y} \geq 0$. Since $b - a + c\frac{B(t)}{Y}$ is decreasing in $t$, we have $b - a + c\frac{B(L(\gamma))}{Y} - 2N(L(\gamma))\frac{b}{m(Y)} > 0$ again. Since $b - a + c\frac{B(t)}{Y} - 2N(t)\frac{b}{m(Y)}$ is decreasing in $t$, the results follow immediately using the intuition developed in the proof of part (i) above, as $\lambda'$ can change sign at most once on the interval $[L(\gamma), T]$. □