Growth and Innovation in Platform Ecosystems[*]

(Extended Abstract – Work In Progress)

Geoffrey Parker	Lones Smith	Marshall Van Alstyne
Tulane	Michigan	Boston University/MIT
gparker@tulane.edu	lones@umich.edu	mva@bu.edu

September 30, 2010

Abstract

A large and growing portion of the economy transpires within software platforms. These environments are bounded high growth economies that embed many novel features: the platform owner sets rules for participation; R&D occurs rapidly, with generations of innovations building on one another; there are profit and adoption tradeoffs in open versus closed code.

This paper introduces and develops a new model of platform growth, inspired by the standard growth literature. At each moment in continuous time, the platform owner must choose (*i*) how much source code to open and make available to a developer community, (*ii*) how fast to absorb developer code into the platform, (*iii*) how much to tax developer output, and (*iv*) how much to invest in the platform itself. Without any public code, the control problem reduces to the growth model of Solow (1956).

Our platform model captures an infinitely recursive R&D ecosystem with innovation spillovers across developers. To optimize growth, the platform owner must internalize this recursive externality. Developers also innovate less the sooner they lose their code, but the platform grows faster with earlier code expropriation. The platform thus willingly privately contributes to the collective public good of open source code, as part of a dynamic profit-maximizing strategy to encourage taxable innovation.

We find that consistent with a variety of data, larger platforms should impose higher developer fees and taxes, expropriate less, and save more. We then compare the platform to a benevolent social planner.

[*]This paper is supported by National Science Foundation grant #0925004.

1 Introduction

A substantial and growing fraction of the economy is represented by platforms — closed economic ecosystems whose owner sustains their development, and who can set rules for opening platforms and for choosing when to bundle downstream applications into the platform. Unlike a standard economic growth model, the platform owner can divest some of his code into the public domain and thereby profit from an external community of developers endowed with commercially valuable ideas that augment this code.

This paper models a platform owner as a landlord to developer firms that build and sell applications on a shared resource. We assume a heterogenous continuum of small developers indexed by the quality of their product ideas. All market power therefore belongs in the hands of the platform. Since platform code depreciates rapidly, the owner must choose how fast to re-invest revenues into building the platform, faced with geometric diminishing returns to its size. These production and savings-investment decisions derive from the standard exogenous growth paradigm (Solow, 1956), in our continuous time setting. At this point, we diverge from this classic growth framework.

In our key modeling innovation, the owner must choose how much of its platform to open up for developers to build upon. So we introduce two legal dimensions of capital — privately-held and publicly-held platform code, which are both nonrival. While the owner has access to his private code, developers can only build on code that has been irreversibly opened to the public. Faced with potential competition from other platforms that may arise, the owner can only charge for the surplus value his private code confers upon consumers. In our continuous time setting, the platform constantly depreciates, and the owner therefore chooses how fast to replenish public domain code.

At a simple level, our paper makes a contribution to the dynamic R&D literature. For we allow two platform choices regarding the developer ecosystem: First, developers may see

their revenues taxed by the platform — as with Apple's 30% rate on iPhone applications. We therefore model the choice of platform "tax rate." Second, unlike firms producing tangible (non-information or rival) goods, the platform owner may at some point copy or confiscate an application that adds value to the platform. We model this simply as a choice of how long application producers may own their products. In practice, a platform owner often creates a functionally equivalent application he then bundles into the platform. This is analogous to a choice of patent length for R&D firms, and captures an intrinsic tradeoff: Bundling applications into the platform sooner reduces innovation incentives, but enriches the platform for future developers. The platform eventually embeds arbitrarily many rounds of innovation sequentially layered on one another (literally a continuum). Our paper therefore tells a much richer story of R&D than usually told.[1]

Our bigger picture is that the platform environment as a closed economy with endogenous growth. Faced with a depreciating code base, the platform sponsor is a social planner who must directly reinvest in its maintenance. In this way, he acts like a landlord who must balance current against future consumption at the margin. His golden rule solution demands greater reinvestment equates the marginal current flow payoffs and the discounted future flow payoffs. For instance, his savings rate is higher with a greater depreciation rate.

Seen through this growth theory lens, we introduce two new tradeoffs in the platform owner's problem. First, he must choose how much of the platform code to open to developers. Intuitively, more public code enriches the production environment for developers, who use the open code resource as an input for their future production. This is especially true with modular programming, in which public code offers more bricks and mortar for future code (MacCormack et al. (2007)). Freely subsidizing additional developers incurs no marginal cost in technology markets since software, design specs, templates, and interface standards are nonrival information goods. In this respect, the platform differs from the landlord — who

[1]The leading papers here Green and Scotchmer (1995) and Chang (1995) explore two stage innovation.

knows that when he allocates land to one tenant, it is unavailable to another. This mirrors the knowledge spillover growth model of Romer (1986). Whereas he focused on the positive external effect of new knowledge that "cannot be perfectly patented or kept secret", we explicitly model the public domain code as a capital stock. But freely transferring resources, however, still incurs an opportunity cost insofar as it reduces direct platform sales. So more openness helps future innovation at the cost of lower current profits. This embodies a classic tradeoff (see Besen and Farrell (1994)).

The platform sponsor then profits from the developer ecosystem in two ways — the tax rate on developer revenues and the finite proprietary period before expropriation. A higher tax rate or shorter proprietary period helps the platform, but reduces developer profits, and thereby this shrinks the developer ecosystem. The platform sponsor selects a positive tax rate and positive expropriation rate that equate his dynamic tradeoff with the developers.

Second, once the platform sponsor learns which downstream applications are successful, it faces a choice. It can bundle successful applications into the platform to increase platform value or it can leave the applications with downstream developers. If it commits to the latter course, developers increase their effort and the value they create. In contrast, if the platform owner bundles downstream applications, developers curb their effort but the platform value increases. This captures the essence of the *ex ante* - *ex post* tradeoff in technology markets. When Microsoft bundled web browsing, instant messaging, streaming media multi-threading, and disk compression into Windows — all features originally developed by downstream developers — venture capital firms reduce funding any startups whose products might be bundled into the operating system. Yet new standardized features increase platform value to consumers, to Microsoft, and even to later period developers. In our model, efficient husbandry of platform resources demands that the platform owner properly optimize the tradeoff between openness and bundling to best solve a public goods problem, coordinate developer activity, and optimize the growth path for platform innovation.

2 The Model

We develop a dynamic model with a mass $K > 0$ of *private code* known only by the platform owner, and a mass $L > 0$ of *public code* that can be built on by others. The public and private code bases evolve with the passage of time. We suppose that code erodes at a constant *depreciation rate* $\delta > 0$. This rate surely exceeds that for standard physical capital (eg. 15%) for two reasons. First, the technological environment is rapidly changing, and software must therefore be quickly re-configured, possibly with new functionality added. Second, the platform software environment is prone to attack by hackers, and this requires constant defensive adaptations. Microsoft updates for Windows operating systems, for instance, often arrive more than once a week in response to the latest discovered vulnerabilities.

Growth models explicitly model preferences and production, and venture that there are diminishing production returns to additional capital (per capita). We study a platform optimization, and therefore need not incorporate consumer preferences. As with the growth literature, we assume a concave production function of the capital level. More specifically, we posit geometric decreasing marginal consumption value of platform code. Namely, total code $K + L$ is worth $\psi(K + L)^\alpha$ to consumers, where $0 < \alpha < 1$. Think of $\psi > 0$ as the economic size of the platform market. Working within the geometric functional family is our critical simplification occasioned by the many additional structures we introduce.

Our public code presents the core substantive economic difference from growth models. A start-up firm could presumably bundle the public code itself, and create a free limited platform of consumption value L^α. We thus assume that the platform owner is constrained by *potential competition* from future platforms that may arise. To best capture this, we assume that the platform owner can only profit from the value-added conferred by his private code. He can thus sell his platform for an annuity value of the difference $\psi(K + L)^\alpha - \psi L^\alpha$. Without public code ($L = 0$), platform payoffs reduce to standard geometric payoffs ψK^α.

The platform owner produces new code. But it is without loss of generality to assume that it only adds to the private code base, and then sustains the public code by "divesting" private code into the public domain. To this end, let $\pi > 0$ be the *divestiture rate* or *publication rate* of private code into the public domain.[2]

Private code has two engines of growth. First, the platform owner can re-invest in its maintenance. Analogous to the Solow growth model, we assume that the platform owner saves a fraction $0 < s < 1$ of his platform revenues to re-invest.[3]

For the second source of growth, we allow the platform owner to write contracts with application developers that confer only a limited duration monopoly on their products. After this exclusionary period, the application may either be folded into the private platform code, or opened to competition. Reflecting this contractual setting with eminent domain, we assume that the platform owner *expropriates* developer code into the private code base at some rate $\rho > 0$. We derive this below, but for now, let us write the developer code produced in the reduced form DL^γ, for some $D > 0$. Notably, this also belongs in the geometric family. In other words, there is an inflow ρDL^γ into the platform code base. As we shall see, the constant D reflects the technology of software development as well as the optimizing behavior of developers to enter and produce code.

The platform owner also takes a "piece of the action" from the applications firms — for instance, Apple taxes 30% of developers revenues for the iPhone and zero for Mac.[4] So motivated, we venture that the platform chooses a *tax rate* $0 < \tau < 1$ on developer revenues.

The platform owner chooses parameters ρ, π, s, τ to maximize the present value of the revenues from direct sales and developer taxes, discounted at the interest rate $r > 0$:

$$(1-s)\psi[(K+L)^\alpha - L^\alpha] + \tau DL^\gamma \tag{1}$$

[2]We suppress the time argument for π and all other variables in our model for notational ease.
[3]For instance, Microsoft has lost money on XBox, having invested more in its creation than it has sold.
[4]Windows charges 0%; Salesforce.com charges 30%; Amazon Kindle charges 70%.

3 The Developer Ecosystem

We now analyze the central role played by the developer ecosystem in sustaining the external returns to public capital. Steve Jobs recently boasted of 225,000 iPhone and iPod touch apps in the App Store, and an excess of $1 billion of tax revenues from developers in the App store.[5] The number of developers for the Windows platform swamps this number. In a modeling assumption consistent with this myriad of firms, we assume free entry from a heterogeneous continuum of potential developers.

The continuum assumption embodies a lack of market power by developers. Developers differ by their ideas $x > 0$, where we envision that an idea simply scales up the value of the developer code. We assume that the mass of ideas x has density $g(x) = \beta x^{-\beta-1}$ on $[0, \infty)$, where $\beta > 2$. Equivalently, the mass of ideas above any level x equals $x^{-\beta}$. The thinness of this tail determines the level of the external returns.

While developers live in a dynamic world, their decision formally reduces to a static optimization. We assume that any application firm with idea x can produce code of value

$$v(x, m) \equiv Axm^{\sigma}L^{\xi} \tag{2}$$

by using a variable input m, costing $wm + \phi$. We might think of the variable input m as hours of engineers, for instance, while the fixed cost $\phi > 0$ might be a building rental, but it can also embed a platform-imposed developer fee.[6] Think of the code value as a flow, owing to the steady-state arrival of customers. On the other hand, costs are treated as upfront, measured in current dollars — as if the developer first rents its facility and hires its engineers, and afterwards sells its product. The coefficient $A > 0$ is a technology parameter, measuring the ease of platform code re-use. It is higher, for instance, with greater software modularity

[5] 2010 Apple World Wide Developers Conference.

[6] Video game platforms like XBox typically charge licensing fees from $3 to $10.

of any applications.

Notably, ξ captures the complementarity of public code to the developer ecosystem. To preclude a dynamic free lunch for the platform, we must discipline the production function: It should be subject to the same diminishing returns to additional public code incurred by the platform as a function of total code. Specifically, we assume that any developer can optimally achieve a returns of the order L^{α}. For the platform itself could always siphon off some of its engineers to form an application firm itself — assuming of course that it had a sufficiently good idea for an application. We will soon see that this condition holds exactly when $\xi = (1-\sigma)\alpha$. And while production is of the Cobb-Douglas form, its homogeneity degree will in general be less than one.

Developers cannot sell code of consumer value $v(x,m)$ for full-price, because consumers can wait until the code is rebundled into the platform. We assume the developers set a price $p(x,m)$ that leaves consumers indifferent about waiting, and buying immediately. Intuitively, a greater expropriation rate blunts a firm's ability to charge for its applications. Given the expropriation rate ρ, consumers expect to discount this future surplus by the factor

$$1 - R \equiv \int_0^{\infty} \rho e^{-\rho t} e^{-rt} dt = \frac{\rho}{r+\rho}$$

Then the market price $p(x,m)$ of developers solves the consumer indifference equation $v(x,m) - p(x,m) = (1-R)v(x,m)$, namely, $p(x,m) = Rv(x,m)$. So, the present value of developer code scales the revenue stream by the effective discount factor $\theta \equiv R/r = 1/(r+\rho)$.

Subtracting the platform taxes, the developer's expected present value of profits is then:

$$\theta(1-\tau)p(x,m) - wm - \phi \equiv \theta(1-\tau)Axm^{\sigma}L^{\xi} - wm - \phi \tag{3}$$

recalling expression (2). Taking first order conditions yields optimal input level m_* obeying

$\sigma\theta Ax(1-\tau)m_*^{\sigma-1}L^\xi = w$. Intuitively, we learn that firms with better ideas are larger. Substituting this into (3), the developer with idea x has the *developer profit function*:

$$(1-\sigma)(1-\tau)Axm_*^\sigma L^\xi - \phi = (1-\sigma)\left[\theta A(1-\tau)xL^\xi\right]^{1/(1-\sigma)}(w/\sigma)^{-\sigma/(1-\sigma)} - \phi$$

We can now see that when $\xi = (1-\sigma)\alpha$, then each developer x has the desired response of code production to the public code L^α. Those firms with the best ideas $x \geq \underline{x}$ enter, and only the marginal one with idea $\underline{x}$ earns zero profits. This yields the expression:

$$\underline{x} = \frac{(\phi/(1-\sigma))^{1-\sigma}(w/\sigma)^\sigma}{\theta A(1-\tau)L^\xi}$$

We see that with a greater public code L, this marginal firm has an even worse idea, and so there is more entry into the developer pool.

Given optimal entry into the ecosystem, the flow of total code produced has gross value:

$$\int_{\underline{x}}^\infty Axm_*^\sigma L^\xi \beta x^{-\beta-1}dx \;=\; AL^\xi\left(\sigma\theta A(1-\tau)L^\xi/w\right)^{\sigma/(1-\sigma)}\beta\frac{\underline{x}^{1/(1-\sigma)-\beta}}{\beta-1/(1-\sigma)}$$

where we have integrated the value coming from all active developers. Observe that this is finite if and only if $\beta(1-\sigma) > 1$. If we define $\gamma \equiv \xi\beta$, then this is the desired functional form DL^γ for the monetary value of the developer ecosystem supply assumed earlier, for the constant:

$$D = \frac{\beta A^\beta(1-\tau)^{\beta-1}\theta^{\beta-1}}{\beta-1/(1-\sigma)}[\phi/(1-\sigma)]^{1-\beta(1-\sigma)}(w/\sigma)^{-\beta\sigma} \tag{4}$$

If the developer revenues equal DL^γ, then exploiting the optimal input level m_*, the *total developer profits* amount to

$$(1-\sigma)DL^\gamma - \phi\underline{x}^{-\beta} = DL^\gamma/\beta \tag{5}$$

after some algebraic simplification. Provided the public code is not too complementary to developer ecosystem code creation, we have $\gamma = \xi\beta < 1$, and thus there are diminishing ecosystem returns to public code. We henceforth analyze the model under this assumption.

Lemma 1 *The ecosystem supply of code has geometric returns with exponent $\gamma > \alpha$, and thus greater than that of any one developer.*

This follows at once from $\gamma = \xi\beta = (1-\sigma)\alpha\beta > \alpha$. More intuitively, this source of additional returns owes to the expansion of existing developers and the entry of new ones.

4 Public and Private Code Capital Stock Dynamics

We now turn to the dynamical system facing the platform. Given the stated depreciation and code publication rates, the public code admits the law of motion:

$$\dot{L} = -\delta L + \pi K \tag{6}$$

Easily, the public code stock would evaporate absent any private code publication.

Likewise, the private code base evolves according to the law of motion:

$$\dot{K} = -(\delta + \pi)K + s\psi[(K+L)^{\alpha} - L^{\alpha}] + \rho D L^{\gamma} \tag{7}$$

The first term represents the losses from depreciation of old code, and maintenance of the public code base by divestiture. The middle term captures the new private code due to savings (reduced profit consumption). The last term consists of the gains from expropriation of the developer code into the private code base.

Finding the fixed point of the system (6)–(7), the ratio of public and private capital stocks obeys $L/K = \pi/\delta$, by stationarity in (6). Just as in the Solow growth model, the

resulting fixed point is stable under the adjustment dynamics. Figure 1 plots the level sets of equations (6) and (7), which cross at the fixed point $(\bar{K}, \bar{L})$ where $\dot{K} = \dot{L} = 0$.

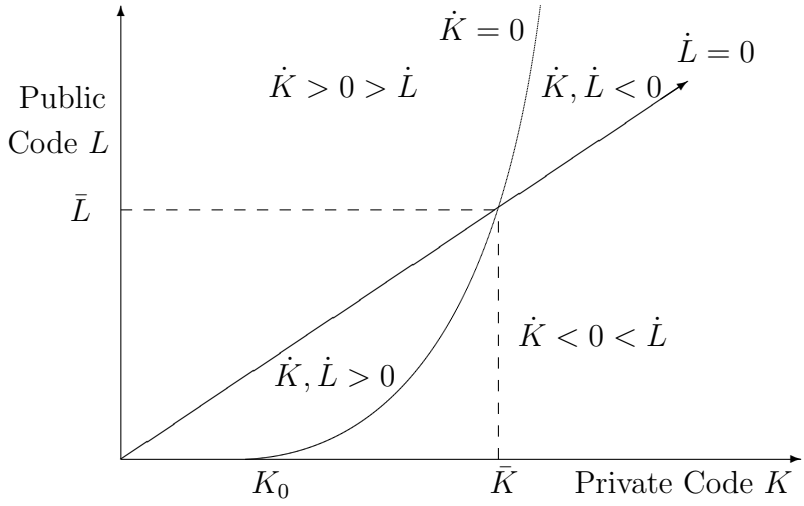

Figure 1: **Dynamical System.**The fixed point for private and public code levels is stable under the adjustment dynamics (6) and (7).

Theorem 1 (Steady-State Private / Public Code) *The fixed point $(\bar{K}, \bar{L})$ is unique, and solves $\bar{K} = \Delta \bar{K}^{\gamma} + S\bar{K}^{\alpha}$ and $\bar{L} = (\pi/\delta)\bar{K}$, for coefficients $\Delta, S > 0$. The stationary private capital $\bar{K}$ rises in s, α and A, and falls in $\tau, \phi, r, \rho, \delta$ and w.*

Proof: One can verify that the $\dot{K} = 0$ curve hits the horizontal axis at the positive solution K_0 of $(\delta + \pi)K_0 = s\psi K_0^{\alpha}$, and initially climbs with a positive slope, as in Figure 1. Imposing $\dot{K} = 0$ in (7) yields

$$K = \frac{\rho D L^{\gamma} + s[(K+L)^{\alpha} - L^{\alpha}]}{\delta + \pi} \equiv bL^{\gamma} + c(K+L)^{\alpha} - cL^{\alpha}$$

implicitly defining the constants b, c. To plot this curve, observe that $K > bL^{\gamma}$ when $K > 0$, but that as $L \uparrow \infty$, the gap $K - bL^{\gamma} = c(K+L)^{\alpha} - cL^{\alpha}$ vanishes. Thus, the curve necessarily crosses upwards any line through the origin, such as $L = (\pi/\delta)K$.

Simplifying using $\bar{L}/\bar{K} = \pi/\delta$, we find that

$$\bar{K} = b(\bar{L}/\bar{K})^{\gamma}\bar{K}^{\gamma} + c(1+\bar{L}/\bar{K})^{\alpha}\bar{K}^{\alpha} - c(\bar{L}/\bar{K})^{\alpha}\bar{K}^{\alpha} \equiv \Delta\bar{K}^{\gamma} + S\bar{K}^{\alpha} \tag{8}$$

for the suggestively-labeled developer and savings constants

$$\begin{aligned}
\Delta &\equiv b(\pi/\delta)^{\gamma} = \frac{\rho D}{\delta+\pi}(\pi/\delta)^{\gamma} = \frac{\rho}{\delta+\pi}(\pi/\delta)^{\gamma}\frac{2\beta}{\beta-2}\theta^{\beta-2}\phi^{1-\beta/2}A^{\beta}(1-\tau)^{\beta-2}w^{-\beta/2} \\
S &\equiv c(1+\pi/\delta)^{\alpha} - c(\pi/\delta)^{\alpha} = \frac{s}{\delta+\pi}\left((1+\pi/\delta)^{\alpha} - (\pi/\delta)^{\alpha}\right)
\end{aligned}$$

There is a unique solution to $\bar{K} = \Delta\bar{K}^{\gamma} + S\bar{K}^{\alpha}$, as seen in Figure 2. This graph makes it clear that any parametric change that raises Δ or S pushes up the map $K \mapsto \Delta K^{\gamma} + SK^{\alpha}$, and thereby inflates $\bar{K}$. It is apparent from (4) that D falls in τ, ϕ, δ, w, r, and rises in A, ρ. Also, S rises in α, S and falls in δ.

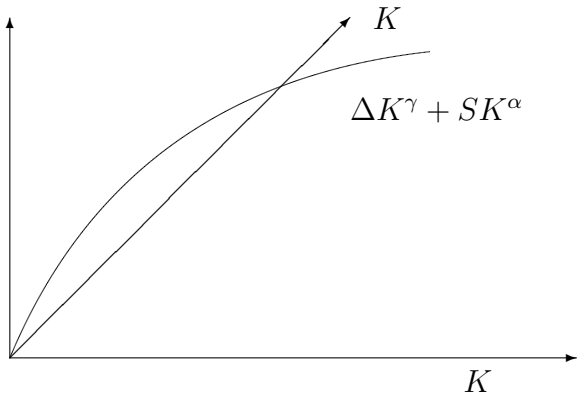

Figure 2: This depicts the logic for a unique steady-state private code capital K, and its comparative statics.

5 The Platform Solution

As with growth theory, a distinction exists between the *Golden Rule* and the *Modified Golden Rule*. The former maximizes the steady-state payoffs, and the latter maximizes the present value of steady-state payoffs, and therefore achieves a lower steady-state payoff (Phelps, 1961; Ramsey, 1928). We shall for now restrict attention to the simpler golden rule.

The Golden Rule is the 5-tuple of savings rate s, tax rate τ, developer fees (embedded in the fixed costs ϕ of the developers), code publication rate π, and expropriation rate ρ that maximizes steady-state revenues (1), subject to steady-state in (6) and (7). But we can make this an unconstrained optimization, by substitution for K, L. The resulting steady-state value becomes

$$V(s, \tau, \phi, \pi, \rho) = (1 - s)\psi \bar{K}^{\alpha}[(1 + \pi/\delta)^{\alpha} - (\pi/\delta)^{\alpha}] + \tau D(\pi/\delta)^{\gamma} \bar{K}^{\gamma} \tag{9}$$

using the implicit definition for $\bar{K}$ in (8). We maximize this value with respect to controls s, τ, π, and ρ.

Theorem 2 (The Platform Solution) *Then its golden-rule optimal savings rate obeys* $0 < s < 1$*, the tax rate* $0 < \tau < 1$ *falls, the developer fees* $\phi > 0$*, and the code expropriation rate is* $\rho > 0$*, and the code publication rate is* $\pi > 0$*.*

Observe the simple implication of the positive code publication rate $\pi > 0$. This is a case with the public provision of a private good (Bergstrom et al., 1985), but driven entirely by the dynamic innovation incentives.

We now venture some key comparative statics predictions of the model.

Theorem 3 (Larger Platforms) *Assume that the platform grows, so that* ψ *rises. Then its golden-rule optimal savings rate* s *falls, the tax rate* τ *falls, the developer fees* ϕ *falls, and the code expropriation rate* ρ *rises.*

Proof: According to the method of monotone comparative statics Topkis (1998), any optimized control variable x^* rises in a parameter ℓ if $V_{x\ell} > 0$ whenever $V_x = 0$. In other words, we must sign the cross partial just when the first order condition holds. This comparative static is easier than others due to the simple multiplicative way that ψ enters into the value expression (9). Thus, its derivative in ψ is

$$(1-s)\bar{K}^{\alpha}[(1+\pi/\delta)^{\alpha} - (\pi/\delta)^{\alpha}] \tag{10}$$

since the last term of (9) is unaffected by ψ. Now, this rises in $\bar{K}$, and so falls in ϕ, τ and rises in ρ, by Theorem 1. The savings rate result is trickier, for it is not clear if (10) rises or falls in s — being the product of an increasing and a decreasing term. We proceed indirectly: Since D and $\bar{K}$ both rise in ρ, the last term of (9) rises in s. Since we have optimized in s, its derivative of (9) vanishes. Thus, the first bracketed term of (9) perforce falls in s. QED

Consistent with this result, Microsoft's XBox platform has yet to make a profit, having "saved" much more than it has earned in platform sales and developer taxes.

6 WELFARE ANALYSIS

The social planner values all the profits of the developers, and not just the tax revenues (1), as well as the entire consumer surplus. But the planner ignores the platform transfer from consumers, and developer taxes. In other words, the planner's steady-state value function is

$$W = \psi(\bar{K} + \bar{L})^{\alpha} + D\bar{L}^{\gamma}/\beta = \psi\bar{K}^{\alpha}(1+\pi/\delta)^{\alpha} + D\bar{K}^{\gamma}(\pi/\delta)^{\gamma}/\beta$$

hereby building on the expression (5) for total developer profits. How how the public code figures so much more prominently in the social planner's objective function than in the

platform owner's. In other words, at the margin, those controls that inflate the public code will be larger. On the other hand, the planner does not care about developer taxes.

Theorem 4 (Social Planner) *Compared to the platform owner, the social planner would choose a zero tax rate τ, a larger public code publication rate π, and would expropriate faster.*

In other words, even though the platform owner is privately providing the public good, he is not doing so to the extent that a benevolent social planner would desire.

References

Bergstrom, T., L. Blume, and H. R. Varian (1985). On the private provision of public goods. *Journal of Public Economics 29*(1), 25–49.

Besen, S. and J. Farrell (1994). Choosing how to compete: Strategies and tactics in standardization. *The Journal of Economic Perspectives 8*(2), 117–131.

Chang, H. (1995). Patent scope, antitrust policy, and cumulative innovation. *The RAND Journal of Economics 26*(1), 34–57.

Green, J. and S. Scotchmer (1995). On the division of profit between sequential innovators. *RAND Journal of Economics 26*(1), 20–33.

MacCormack, A., J. Rusnak, and C. Baldwin (2007). The impact of component modularity on design evolution: Evidence from the software industry. Harvard Business School Technology and Operations Mgt. Unit Research Paper No. 08-038.

Phelps, E. (1961). The golden rule of accumulation: a fable for growthmen. *The American Economic Review 51*(4), 638–643.

Ramsey, F. (1928). A mathematical theory of saving. *Economic Journal 38*(152), 543—559.

Romer, P. (1986). Increasing returns and long-run growth. *The Journal of Political Economy 94*(5), 1002.

Solow, R. M. (1956). A contribution to the theory of economic growth. *Quarterly Journal of Economics 70*(1), 56–94.

Topkis, D. M. (1998). *Supermodularity and Complementarity*. Princeton University Press.