Global Software Users' Co-op

Mark Latham

This revision: April 3, 2016
Original version: November 9, 2012
Future drafts at votermedia.org/publications

Abstract

Software users can get a better deal on some software by organizing into a large buying group. Creating a consumer cooperative is a promising way to do this. Large group purchases can help solve problems in the software business, including monopoly power, the costs of intellectual "property" law, and deadweight loss from prices exceeding marginal costs. For such a co-op to work efficiently, it needs an institutional framework that will minimize agency costs and enhance competition among software suppliers.

A piece of software, once created, can be copied and used by any number of people with no additional production cost. For that reason, the economics of software resemble the economics of public goods provided by governments, which benefit an entire community at once. So designing a framework for the software business is similar to designing a framework for supplying public goods. Ideas for either of these may help in designing the other.

"Votermedia" is a new competitive system that lets voters choose public goods providers in a democracy. It has been developed and tested for the past six years in the University of British Columbia's student union. It can be adapted to let software co-op members allocate their pooled funds by vote among competing software suppliers.

This paper outlines the institutional structure, voting system and rules for such a co-op. Software providers would compete to sell bundles of software (broadly defined to include any information goods). Co-op members would get usage rights for all bundles provided, and would vote to allocate funds among (but not within) the bundles. Allocation of funds among components within a bundle would be decided by contracts between bundlers and developers.

If votermedia proves successful for buying software, that will further validate its design, and speed its adoption in more democracies. Allocating even a small fraction (less than 1%) of a democracy's budget via votermedia, can support investigative journalists and public policy critics that create net public benefits by reducing agency costs in government. Votermedia can likewise be implemented in corporations, to improve director accountability, financial performance and social impact.

Mark Latham is a financial economist, and founder of Votermedia.org (cv: linkedin.com/in/marklatham; email: mark[at]votermedia.org) – comments invited.

Contents

1. Introduction: How should we pay for shared benefits?

There are many types of non-rivalrous goods -- goods that can be used by many people at once, where additional users cause no additional production cost and no reduction in benefits to other users. Examples include street lighting and digitized music. This paper will focus on two simplified archetypes of non-rivalrous goods, here termed "public goods" and "commercial software". Although typical usage of these two terms includes a wider range of cases than our definitions below, the narrowed focus is convenient for exposition. So for most of this paper, we will limit these terms to non-rivalrous goods with the following features:

Public goods --
(a) provided by governments to citizens;
(b) funded by taxes;
(c) non-excludable -- there is no practical way to prevent anyone from using.
Examples: street lighting; national defense.

Commercial software --
(a) provided by businesses to retail users;
(b) funded by user payments;
(c) excludable -- there are practical ways to control who uses.
Examples: digitized music; Microsoft Office.

While many non-rivalrous goods can be classified into one or the other of these two archetypes, one important good spans both: journalism. Some of the benefits of journalism are non-excludable (e.g. reduced government corruption), but some are excludable (e.g. being entertained). Mainly for that reason, some journalism is funded by taxes while some is funded by user payments.

This paper will examine the social welfare impact of different possible institutional frameworks for producing and using non-rivalrous goods. The frameworks may affect:
- which goods are produced;
- who produces them;
- who gets paid how much for producing them;
- who uses them;
- who pays how much for them.

Our existing institutional frameworks for public goods and commercial software seem to fall far short of any social ideal. Governments supplying public goods suffer substantial agency costs, manifested by inefficiency and corruption. The commercial software business suffers different but likewise substantial inefficiencies, including:

(a) Deadweight loss -- Any price charged to users will be higher than the software's (zero) marginal production cost. Some potential users may value the software less than its price, and thus not buy it. Their lost benefit is a net social loss since marginal cost is zero.

(b) Costs of an intellectual "property" (IP) legal system[1] --
- creating and defending patents;
- obstructing the creation of new inventions that overlap existing patents to some (perhaps minor) degree;
- lobbying for expansion of IP protections beyond the social optimum;

1. Lessig (2006, Chapter 10) gives a detailed discussion of the challenges and tradeoffs in designing an IP law system.

- lost benefits of software that is not created because of imperfect IP law protection.[2]

(c) Costs of gaining monopoly power (besides the IP system costs above), such as building dominant market share via first-mover advantage, advertising or mergers.

(d) Transaction costs of organizing many small payments for usage rights.

(e) Costs of creating and using systems for granting access to payers while excluding non-payers. Even payers may have only limited usage rights, in spite of zero marginal costs of relaxing those limitations.

To remedy these inefficiencies, various reform strategies have been pursued. For governments supplying public goods, democratic reforms can be seen as strategies for reducing agency costs by making elected leaders more accountable to citizens' interests. Without surveying the wide range of democratic reform projects, a few examples are Rootstrikers.org, UnitedRepublic.org and FreePress.net.

Among strategies for reducing inefficiencies in software markets, impressive progress has been made in organizing and encouraging the creation of free software (copyable and usable at no charge). Leading examples include Wikipedia, the Linux operating system, and citizen journalism. Prominent supporting organizations are the Free Software Foundation and Creative Commons. The generosity of free software developers is augmented by the generosity of financial donors, including crowd-sourced donations via such websites as Kachingle for supporting web content, and spot.us for supporting journalism. However, the growth of free software is naturally slowed by human self-interest -- the limited supply of altruism.

Commercial software markets have evolved in ways that alleviate some of the inefficiencies outlined above. Growth of the internet has reduced many transaction costs. Some of the social cost of deadweight loss can be reduced (and captured as profit) by price discrimination -- charging different prices to different classes of users (e.g. students vs home users vs business users). If sellers can sort out types of users who value the software differently, they can charge each user type a price just below its valuation.

Another way to capture deadweight loss is by bundling many software products together, and selling them to users for a single package price:

> "Large digital collections are increasingly common as much Internet content moves from free to fee-based systems and as new forms of digital content, such as satellite radio, emerge. Consider the bundles that constitute XM radio, Cable TV, AOL content, Rhapsody music, Consumer Reports reviews, JSTOR academic articles, and Microsoft Office." (Brynjolfsson & Zhang 2006, p102)

The economics of software bundling are well explained and explored in Bakos & Brynjolfsson (1999; 2000). A simple example is given in section 6 below. Limitations of this strategy include losing the price discrimination potential of charging more to those who use more pieces of software, and needing to guard against monopoly power of very large bundlers.

We could consider supplying some software the way we supply public goods -- via our governments. Software funded by taxes could be used by everyone at no charge, thus eliminating deadweight loss. The reason governments do not fund substantial retail software development is probably that inefficient administration would outweigh the potential social welfare gains, compared with providing excludable goods via commercial markets. (Imagine a government department deciding which software to buy and

2. As Brynjolfsson and Zhang (2006, p101) summarize: "Thus, the debate centers on who will be impaled on the two horns of the dilemma: should creators be deprived of the rewards from their creations or should users be deprived of goods which cost nothing to produce? Either approach is demonstrably suboptimal. It would seem impossible to have both efficiency and innovation when it comes to digital goods. Improving one goal appears to be inextricably intertwined with hurting the other goal."

how much to pay for it.) We do fund some information goods by taxes, but these tend to have substantial non-excludable benefits: public interest journalism and some basic scientific research. Journalism can improve the quality of government (non-excludable), and basic research can serve as a foundation for subsequent applications that may be so many steps removed that the links are difficult to price and charge for. Another limitation of government funded software is the lack of a global government.

Fisher (2004) proposed U.S. national tax funding for music and movies. He reduced the problem of inefficient government administration by automating and crowd-sourcing the fund allocation process. Internet systems would track consumer usage of each piece, and creators would be compensated accordingly. This design has considerable appeal, but has not come close to implementation. There would inevitably be winners and (especially) losers in the entertainment industry from such a plan, so the losers would resist it. It would seem to require a major political push, which has so far not occurred.

Bakos & Brynjolfsson (1999, p1625) mentioned another funding strategy:

> "...a consortium or club of consumers could purchase access to a variety of information goods and make them available to all members for a fixed fee. Some user groups and certain site licensing arrangements for software resemble this approach."

This mechanism has not spread to a significant fraction of the software market, perhaps because the agency costs of such clubs may outweigh potential gains, compared to achieving similar bundling via private sector competitors. Indeed, Latham (2012b) finds evidence of substantial agency conflicts in cooperative organizations.

This paper proposes creating a software users' cooperative with a new governance structure designed to reduce agency costs. As outlined in section 2 below, members would allocate their pooled funds by vote among competing software suppliers. Section 3 details a vote aggregation algorithm (with a model for voter behavior in the Appendix). Rules for software sellers are proposed in section 4, which recommends a laissez-faire approach -- let sellers offer whatever software on whatever terms they choose, and let members vote for what they prefer. Section 5 suggests several tools for helping members make such complex funding allocation decisions, by collecting usage statistics and sharing expert assessments of software benefits.

Section 6 gives examples to illustrate that the co-op will give substantial benefits to members compared with individual retail purchases, taking into account that the transactions must also be attractive to software providers. Stages of potential future evolution and growth of the co-op are considered in section 7. Section 8 explores political implications of a global democratic software co-op, including potential influence on information technology law and regulatory policy. The co-op's voter controlled funding system may spread to democracies and corporations, reducing their agency costs and creating competitive markets for public goods.

2. Proposal: Create a software users' co-op

If we can reduce the agency costs of a consumer cooperative, software users should be able to capture the benefits of reducing both deadweight loss and monopoly profit. Competition to sell to a large group can help negotiate software prices down. Because the potential benefits are large, there are probably many ways of organizing this that would work.[3] Below is a specific proposal, which can no doubt be improved upon.

The usual governance structure of a co-op is for members to elect a board which hires the senior staff. The

3. I would say that a software co-op "works" if it is attractive enough to buyers and sellers that it grows to more than 100,000 members and is generally perceived to increase social welfare, such as by capturing deadweight loss.

staff manage the co-op with board oversight. As reported in Latham (2012b), agency problems at co-ops seem broadly similar to those in democracies and corporations, which likewise elect leaders to hire and oversee managers. Most voters lack the time and expertise to monitor the leaders and staff carefully. As a result, voting decisions do not exercise the necessary discipline on self-serving and underperforming boards.

Latham (2012a) describes six years of test implementations of a new voter information system called "votermedia" which seems to inform voters well enough to reduce agency costs in a democracy. It lets voters allocate a small budget (less than 1% of the organization's total budget) among competing information providers. Competitors provide any information they think voters will value, typically including investigative journalism, policy critiques, election coverage and candidate endorsements. By adding this system to an existing standard governance structure (board elections etc.), accountability seems to increase and thus agency costs decrease.

If we implement votermedia in a software co-op, we can use it not only for voter information, but also for other information goods -- software. We can let members control the allocation of their funds among competing software providers. The same competitors can fill both roles -- software sellers and governance monitors. As long as we maintain enough competition by funding multiple providers (e.g. about ten), they should check and balance each other well enough to minimize agency conflicts, even with these dual roles.

A specific concrete proposal can help focus this discussion. Consider the following design:

> (a) We create a software users' co-op with a conventional governance structure (elected board etc.), augmented with bylaws establishing a votermedia competition for paying software providers.
>
> (b) Each member pays $5/month. This is enough to buy a significant amount of software, especially with the strength of large group purchasing. Yet it is cheap enough for new members to easily risk on a new purchasing model. We will see what we can get for $5. Later we can create optional higher payment levels for buying more and better software.
>
> (c) Any software provider can enter the co-op's votermedia competition. (We might charge an entry fee by paying an entry fee to deter ballot clutter from non-serious entrants.)
>
> (d) Competitors are expected (but not required) to provide software and/or other information goods, including monitoring the co-op itself with policy critiques, election coverage etc. Competitors may develop the software themselves and/or contract with software suppliers. (See section 4 below for detail on the laissez-faire policy for competitors.)
>
> (e) All members get access to all software provided by the competitors. Ideally we should build a system to let a single member login give seamless access to all provided software. This may be something like the "login via Facebook" system used by many websites.
>
> (f) Competitors need not give *exclusive* use of any software to co-op members. Non-exclusive usage rights are sufficient. The co-op would be an additional sales channel for sellers of login-protected software.
>
> (g) Co-op members can vote at any time on how to allocate co-op funds among the competing software providers, on a ballot similar to votermedia.org/ubc. For each competitor, a member can choose to vote for a funding share from this menu: 0%, 5%, 10%, 15%, 20%, 25%.[4] Competitor shares are calculated daily, based on the latest vote received from each member. These funds are

4. Each competitor has a row in the ballot table with their name (linked to their website) and links to recent posts on their blog. A software seller could give a website link that leads to a page listing the software they are providing. Their blog could include news about software they have recently added.

accumulated and paid out monthly. Section 3 below gives details on the vote aggregation algorithm, which is not a simple average, but more like a median. Its parameters are occasionally adjusted so that roughly the top ten competitors get some funding. The rest may choose to continue competing, or drop out, or merge with other competitors.

(h) Co-op staff administer the membership system, accounting, and software infrastructure for logins and voting. The budget for staff and administration could be determined by member vote, in the same competition ballot with the software providers.

Open dynamic competition will determine what software we get for our total budget. Even though we don't all want the same software, we buy a bundle of bundles so that (hopefully) every member gets some software they want. As a bonus, we also get "free" use of a wide range of other software we wouldn't have paid full retail price for, but might benefit somewhat from using. This is the gain of recapturing deadweight loss. Other members have paid for it, so the developers are compensated enough compared with their normal retail sales channel. More detailed analysis of benefits is in section 6 below.

3. Vote aggregation

The algorithm that aggregates member votes (to determine software sellers' funding shares) is crucial to the co-op's success. This section outlines the reasoning behind the votermedia system used for six years in the University of British Columbia's student union. We have no proof that this system is optimal, but it seems to work well. Although they were low budget experiments (typically $8,000 per year) for funding student journalists, the economics of aggregating voter interests are similar to the case of funding a broad range of software. Further evolution will no doubt be desirable, but this can serve as a starting point for designing the co-op's voting system.

Especially because the co-op will involve so many participants in various interacting roles, it is hard to forecast how this new game will play out. We can expect to learn far more from early implementation than from prognostication, so we should "release early and often."

A conceptual starting point for a competitive fund voting system is the majority vote often required to approve a large municipal spending decision (e.g. building a bridge). Such a proposed project has an estimated budget, citizens vote yes or no on the proposal, and the majority decision is followed.

Suppose instead that we want to approve a budget allocation for a project with a flexible budget, while letting voters decide the budget. Applying the median voter theorem, we could let each voter specify the budget they think best, and aggregate those votes by choosing their median. Under the mild assumption that each voter's preferences are single-peaked, the median would be chosen by a majority in a two-choice vote versus any other budget level. Also there is no incentive for any voter to lie (i.e. to specify a budget other than the one they think optimal). In contrast, using the mean instead of the median would create incentives for voters to lie by exaggerating their deviations from the mean.

So the co-op could conduct a separate vote on each software supplier to decide how much to pay them. However, we would need to somehow make sure that our total budget for software is not exceeded. At UBC we adjusted the cutoff up or down from the median (= 50^{th} percentile), to the percentile that would match our budget. We treated non-votes as votes for a zero allocation, in cases where someone voted funding for some competitors but didn't vote on other competitors. Combining these two principles, they are equivalent to the following rule: Find the cutoff number of votes such that, when we award each competitor the funding level "approved" by the cutoff number of votes, the awards add up to our total budget. A vote "approves" a funding level if it is for that level or higher. So a vote for 10% would count as approval for an award of 9%, but it would not be approval for an award of 11%.

For the first few years at UBC, voting was for dollar amounts (menu options like $500, $1000 etc.), but later we changed to percentages of the total award pool (5%, 10% etc.). This change was especially convenient when we implemented a continuous voting system, with daily awards and a total budget flow that could change daily.

We decided that choosing from a few discrete menu options is preferable, because it is easier for voters than typing in any amount they choose into a response box. However, the cutoff algorithm described above would then result in awards that respond discontinuously to gradual shifts in voting, jumping from one menu level to the next. So we interpolate each vote -- for example, a vote for 10% would be spread uniformly from 7.5% to 12.5%.

In the first three years of votermedia at UBC, the journalism competition was focused on covering the annual election of student union leaders in January. Competitors covered the election campaign, and students allocated shares of the $8,000 award pool by voting on a newly appended votermedia section of the election ballot. In later years we developed a continuous-time online voting system (at votermedia.org/ubc) to allocate some funding throughout the year. We adapted the voting rules and aggregation as follows:

(a) Voters can vote at any time.

(b) Awards are calculated daily, based on the latest vote from each voter on each competitor.

(c) Because it is unrealistic to expect most voters to vote daily, each latest vote continues to be counted even if it is several days old. To reflect the fact that competitor performance and voter opinions change gradually through time, we let the weight of each vote decay through time. The decay schedule we have been using keeps the weight at 100% for 10 days, then diminishes linearly for 50 days, dropping 2% weight per day. So votes older than 60 days have zero weight, i.e. no longer counted.

A simple model for voter behavior under the above rules is given in the Appendix.

We made one further adjustment to the aggregation algorithm, to increase competition among multiple information providers. There is a natural tendency for busy voters to pay more attention to the top ranked competitors. Indeed, the votermedia system is designed to enable this type of behavior, where busy voters benefit from the work of those voters who have time to compare the relative merits of more competitors. But this may cause a bias favoring the top ranked competitors, awarding them more funding than is in the best interests of all co-op members. Maintaining competition is important for reducing monopoly profits.

At UBC we tried several methods of ensuring that funding would be spread across "enough" competitors. The first year, we did not let voters determine how the award pool would be divided. Instead, the contest oversight committee sliced the $8,000 pool into eight awards, from a first prize of $1500 to eighth prize of $500. Voters simply checked a box next to each competitor they felt deserved funding, and prizes were awarded in order of vote count (an adaptation of approval voting).

In later years when we let voters determine each award size, we put an upper limit on votes (and thus awards), typically 25% or 30% of the total pool. However, this often resulted in two or three popular competitors hitting the maximum, while it was still difficult for other entrants to get funded at all. It reduced the incentives of the top competitors. It could be compared with having an income tax schedule with zero tax up to a maximum income level, and then 100% tax above that. Pursuing the income tax analogy, we decided that a graduated tax schedule could better serve the goals of spreading the wealth without dampening incentives as much.

So we chose a single parameter exponential function to scale down each competitor's number of votes:

after-tax vote count = (pre-tax vote count) / (spread ** share)

** means exponent, to the power of

"share" is the competitor's current award share as a decimal, e.g. 27% would mean share = 0.27

"spread" is a parameter chosen to give the desired amount of spread-the-wealth effect. spread = 1 would mean no effect. I propose spread = 16 for the software co-op. So for example, if share = 25% and spread = 16, then 1 / (spread ** share) = 0.5, so half the votes are taxed away. To earn the next 1% share would thus take twice as many votes (for shares over 25%) as it would take an unfunded competitor (in votes for shares over 0%) to get their first 1%.

We can also cap the voting menu options at 25% or 30%, just to make sure. The tax schedule will phase in a dampening tendency before the cap is reached. For an example calculation (one-time version -- no vote decay), see votermedia.org/misc/1VMSharesCalcExample.xls.

I would estimate an ideal number of funded software suppliers to be about ten, to maintain a healthy degree of competition. This will require monitoring of the co-op's performance once it gets underway. Statistical analysis of voting and funding shares through time can help us adjust the spread parameter and the vote decay schedule.

Some competitors may not get funded at all, and may choose to drop out or merge with others. The voting system gives a strong incentive for sellers to group into a limited number of competitors, depending on how many brand reputations the voters are willing to assess and share.

Requiring majority votes means that the co-op buys bundles of software in which most members find some useful software. Many of the individual pieces of software can have narrow markets, as long as they are bundled with other software so that in combination the bundle has broad appeal. The parallel in politics is a rainbow coalition of minorities who, together, form a majority.

Software sellers will each appeal to a majority of members, where any pair of majorities must overlap to some degree. Voters are not segmented in opposition to each other as in conventional party politics. Tyranny of the majority is less of a problem in a co-op than in a democracy, since co-op membership is voluntary. Experience will show whether it may be beneficial to have multiple co-ops serving different market segments, although this could undermine some of the pooling and sharing benefits.

4. Laissez-faire for software sellers

To minimize agency costs and administration costs, the software co-op's institutional design keeps democratic control in the hands of members wherever practical. The primary mechanism of democratic control is the voting system that allocates co-op funds among competing software sellers. Any rules empowering co-op administrators to disqualify competitors, would shift power away from members. Therefore such rules should be minimized. If a competitor is somehow detrimental to the co-op, then members can vote not to fund that competitor. Administrators (and anyone else) can advise members to do that, but the decision power should reside with the members. Thus competitors have a strong incentive invest in their reputations for serving co-op members.

To clarify this democratic principle, this section outlines a wide range of information goods, their attributes and delivery methods that competitors could use to appeal to voting members. To explain how and why

democratic control can work well, the next section describes some tools that could help voters make wise funding decisions.

It is important to distinguish between expectations and requirements. For example, we expect competitors to supply software, but we do not require it. Thus competitors may (*or might not*):

- supply information goods such as downloadable software, software usable via the web, journalism, music, movies, other entertainment, telecom services;
- stop supplying some software that they were supplying;
- announce plans to supply software in the future, perhaps contingent on receiving some level of co-op funding;
- change their plans to supply software, with or without giving reasons for the change;
- supply benefits other than software;
- develop software themselves or buy it from others;
- negotiate any form of contract with software developers;
- be for-profit or non-profit organizations;
- disclose their accounting statements, names of owners and developers, developer contract terms;
- supply software that works in only one or more operating systems;
- supply software that is or will become open source or free in any sense;
- require login access to their software;
- require users to give some personal data in order to access software;
- have advertising in their software;
- display their brand logo when software they supply is being used;
- charge user fees for their software in addition to receiving co-op funding; (equivalently) negotiate discounts for co-op members on some software user fees; this freedom may permit expansion of co-op benefits to include those with positive marginal cost;
- allow co-op members to use their software on only one device, or on several devices;
- supply software that another competitor is also supplying to the co-op;
- give a refund on software that a co-op member has already paid for outside the co-op;
- enter more than one team/brand in the co-op competition;
- supply software used by the co-op organization itself, such as voting infrastructure, authorization systems, accounting, tools to help voters (see next section below);
- critique the board, management, policies and rules of the co-op; endorse candidates in board elections;
- critique all competitors supplying software to the co-op; review their software; recommend how much funding share to vote for each;

- host co-op member discussions.

If co-op members want to consider restricting competitors' freedom on any of the above points (e.g. banning software with ads), I would recommend first postponing such a restriction during a trial period, to see if it is really necessary or desirable. The incentive for competitors to please members should be enough to keep negative attributes (like annoying ads) limited to a level where they are outweighed by their positive impacts.

5. Tools to help voters

For the votermedia competition to work well in serving the interests of co-op members, they need to vote funding to the "better" competitors. Of course, not all members have to vote. Those who prefer the simplest involvement in the co-op can just pay $5 a month, and use the software chosen by those who do vote. This section examines the complexity of software funding decisions, and suggests some tools to aid and simplify those decisions.

Deciding which software sellers to fund is a complex task, for reasons including:

(a) Frequency of software use may not correlate well with the amount of user benefit.

(b) As the co-op grows, it will supply more and more software to the point where most members never use most of the software.

(c) The quality of software currently provided by a competitor may not accurately indicate the quality of additional software they would provide with more funding.

(d) A software user may not recognize the degree of competition in each software market. In some markets with substitute products, price can be negotiated down close to production cost. Others markets may be more monopolistic, so price may be closer to user benefit than to production cost. Some software inherently has a high benefit/cost ratio regardless of who develops it.

(e) There may be interactions among competitors, such as sharing source code and other innovations. Members may not be aware of how much valuable code each competitor contributes.

(f) Some members may not realize that some competitors are helping the co-op by advising the decisions of other members voting in board elections. You know about direct benefits from using software or advice yourself, but it's harder to know about indirect benefits you receive from other people using it. Contributions to the co-op's infrastructure software may also be hard for members to evaluate.

(g) Some impacts of software are hard to assess, such as on privacy and security.

(h) Some software sellers may behave in ways that affect the degree of competition in software markets but are hard to observe, such as by legal threats or secret agreements.

Of course, users buying software in conventional retail markets face most of the same challenges anyway. Those imperfect individual decisions would be a more appropriate benchmark for co-op decisions than some unattainable ideal. A well designed collective decision-making process may allocate software funding more wisely than the sum of independent decisions.

Since co-op members are all buying the same software by collective decision, they will have a strong incentive to share insights on which software is useful, to educate fellow members and thus benefit from

their improved voting decisions. They will have two incentives to learn -- to vote better and to find which software is useful for them. So we can expect active member discussions on the full range of co-op software.

Each software seller will of course promote their own software bundle to co-op members, to get more voting support and thus more funding. Such promotion will of course have self-serving bias. But as suggested in section 4 above, competitors may also provide reviews of their competitors' software. If there are enough competitors (perhaps more than seven), then the potential for gaining member voting support as a reward for useful software reviews, is likely to outweigh the bias toward simply denigrating the competition. Reviews are likely to distinguish one competitor from another, even if the reviewer's own software is rated highest. Thus expert software reviewers can get paid for helping co-op members judge the marginal value of more funding for each competitor, taking into account the complex factors listed above.

The co-op's board and staff might also contribute to these important comparisons. Although the co-op's democratic governance design does not give its leaders power to allocate funds among competitors, they can be expected to have expertise worth sharing. Especially if members can vote to determine the staff budget, there will be an incentive to help members in whatever way possible, especially on the key decision of allocating funds.

Besides informing members, all this active discussion among experts and users will also be valuable feedback to software developers, and thus an extra incentive to sell via the co-op. Another benefit: makers of software selected for funding by member vote could brag about that in their retail market promotions.

An important input into software purchasing decisions is the amount of usage the software gets. The co-op should gather usage data and share it with members, sellers and developers. Each member may want to vote funding based on their own usage statistics, and/or on the statistics for all members combined. Sellers who bundle software from many developers are likely to pay each developer based on their software's usage rates. Voting data should also be published, especially the vote counts for those competitors who are not currently funded, since it can help determine their value in a merger or buyout.

Software usage can be measured by number of downloads, by the number of logins for online content, and better still by the amount of time spent using it. Usage time of downloaded software can be measured and reported by code embedded in each application, for those co-op members who agree to allow such reporting.

We can compare other proposed and implemented designs for linking usage statistics to payments for software. Kachingle.com lets individuals donate $5 per month to support free web content, by allocating the $5 among the donor's selected websites in proportion to the number of days that month the donor visited each website.[5] As mentioned in section 1 above, Fisher's (2004) proposal for U.S. national tax support of music and movies would allocate funding according to consumer usage.

Brynjolfsson & Zhang (2006) proposed a statistical sampling method for estimating user demand for each separate software component in a bundle: If the number of users is large enough, it can be worth the cost to offer a random sample of users the option of receiving a cash payment to forgo the use of a specified software component in the bundle. Response to a series of such options will reveal the user demand schedule. If a software co-op grows large enough, such experiments may become worth their cost. The statistical results could be published to help bundlers decide how much to pay for each software component.

5. When I checked at kachingle.com on 2012-11-05 to reconfirm their allocation algorithm, I discovered they had just shifted their business model in October 2012 away from donation for non-excludable content, to focus on payments for access to bundles of excludable (login-protected) content and software. Now it seems they charge a weekly price ($5 or $10) for each bundle they offer, and allocate funds across software within each bundle according to usage rates.

However, usage statistics and user demand schedules do not capture all the relevant benefits to co-op members from the various possible contributions of each developer and bundler, because of the complex factors listed at the beginning of this section. How can co-op members take all these factors and information inputs (discussions, expert advice, usage statistics) into account to vote intelligently on funding allocations? The task seems too daunting for a typical busy individual software user. Especially if voting is optional, wouldn't most members save themselves the trouble and not bother?

Various tools can be developed to make it easier for members to vote funding intelligently in their own best interests:

(a) The simple approach of voting funding shares based on each member's subjective estimates of benefits from each bundler may be accurate enough. With the aggregation of many members' judgments, the wisdom of the crowd may take sufficient account of the complex ways that software providers affect members' interests.

(b) At each member's option, that member's usage statistics (e.g. for the past month) could be displayed next to each competitor on their ballot, as a guide to voting.

(c) As described above, software providers would have an incentive to give advice on how funding should be allocated. Co-op staff may also advise. Members can read advice, choose one or more advisors they find convincing, and follow their advice.

(d) The co-op could build a system to let members automate their voting to copy the advice of their favorite advisor.[6] Such advice could use that member's usage statistics as a factor affecting the shares voted.

(e) Any of these systems should maintain the principles of maintaining member control, and preventing vote-selling by keeping individual votes confidential.

The co-op's institutional features outlined in sections 2 through 5 should create a dynamic competitive market for software, open to entry by new competitors, with control in the hands of software users rather than administrators. As UBC student councillor Alex Lougheed observed: "One of the great things about [votermedia] is that it's self-regulatory. It's a very free market approach to solving a lot of the problems with media today."[7] With the power of sharing large group purchases, this should give software co-op members considerably more benefit than they would get for $5 per month in conventional retail markets.

6. Examples of co-op benefits

For a software co-op marketplace to work, it must be attractive to both buyers and sellers. This section gives some examples to illustrate how the two sides' interests can be balanced to find transactions that will benefit both.

The co-op will provide large scale market organization that reduces transaction costs for users and sellers, and marketing costs for sellers. Users can buy many software products with one transaction, and software providers can sell to many users at once. Many small payments are merged into fewer large payments.

The co-op will also change the economics of each transaction through bundling, collective user choice, and competition. Consider this classic example of how bundling can help recapture deadweight loss. Suppose there are two users and two pieces of software which can generate the following dollar values of benefit:

6. This is similar to a proposal in Latham (2007) for proxy voting of corporate shares.
7. See video "What is VoterMedia?" at votermedia.org/videos/1.

	Software A	Software B
User 1	$10	$3
User 2	$3	$10

Suppose that production costs and competitive conditions are such that the market prices of A and B are $9 each. Then user 1 will buy software A, user 2 will buy software B, and each will get a net consumer surplus (= benefit - cost) of $1. The potential $3 benefits are not received, even though the marginal cost of producing an extra unit of the software is zero. Hence the term deadweight loss, since in an ideally coordinated world, social welfare could have been higher by $3 + $3.

Instead, if A and B are sold as a bundle for, say, $11, then both users would buy the bundle. Buyers and sellers would all be better off in this bundling scenario than in the separate sales scenario above. Each user gets a surplus of $10 + $3 - $11 = $2, and the software makers get revenue of $22 instead of $18. There is no deadweight loss.

This happy win-win story becomes clouded, however, if there is also a user #3 who values both A and B highly enough to buy both of them separately for $9 each:

	Software A	Software B
User 3	$10	$10

Bundling would give user 3 an excellent bargain, receiving $20 of benefits for the bundle price of $11 instead of the separate prices of $9 + $9 = $18. Bundling becomes much more advantageous for users on average, but sellers would receive less revenue (3 x $11 = $33) than with unbundled sales (4 x $9 = $36), so would choose not to bundle.

This example suggests that bundling tends to benefit users more than sellers. Sellers may benefit if demand for multiple pieces of software is spread fairly evenly across users, but bundling may hurt seller revenue if subgroups of users differ substantially in the number of software products they would buy separately.

The co-op proposed in this paper lets users buy one large bundle of bundles, while making bundlers compete with each other to provide better software at limited cost. Some software providers will find this market structure unattractive, while others will be willing to participate. The main determining factor is likely to be the degree of competition and substitutability in their software niche. Examples below illustrate why.

Some software is unique, while some is substitutable with other software. Among music for example, a new hit single from a pop music star would tend to be unique. Fans want to hear that song performed by that artist. An orchestral music recording, however, is likely to have close substitutes. Classical compositions are old enough to be in the public domain, and even among top international orchestras, there are typically more than five who could create recordings of comparable quality.

So the pop music star could price her song as a monopolist (subject to the limits of imperfect protection from copying) and extract super-normal profits. But orchestras will tend to see their revenues competed down to a level sufficient to compensate professional musicians of their stature, not substantially more than that.

Most computer software is substitutable, not unique. A substitute for Microsoft Office, for example, could

be built by some other team of developers.[8] In spite of that, Microsoft probably earns profits on it well beyond its production and marketing costs. There are several reasons for this, primarily related to being a natural monopoly. The large fixed costs of creating and marketing a substitute make it difficult to challenge Microsoft's position.

While Microsoft Office is a leading example in a powerfully dominant position, there are lesser degrees of dominance in most retail software markets. Many software product categories have a leader with most of the market share, and one or two challengers with smaller but significant shares, for example in the range of 5% to 30%. These competitive situations offer the potential for bundlers to contract for software usage rights from challengers at attractively low prices.

Consider the following scenario: The software co-op has grown to 500 members, collectively buying a range of software with their $5 per month, from competing bundlers. A bundler is negotiating with a developer to include their software product in a bundle being sold to the co-op. Its retail price is $50 (per year), and this product has a 10% market share in a product category that is purchased by 20% of all software users. How much payment might the developer require to make the deal worthwhile?

Suppose the developer considers the co-op's 500 members to be a typical cross-section of the user market. Then 20% of them (100) use this category of product, and 10% of those (10 members) are already using this specific developer's product, paying $50/year each on the retail market. If the developer sells it to the co-op, he will lose that retail revenue (10 x $50 = $500). While various other factors might influence this negotiation, a simple model could say that as long as the developer receives at least $500, then the deal is worthwhile.

That would be an amazingly good deal for co-op members -- only $1 each to use a $50 piece of software. There are several ways to assess the benefits. One simple way is to suppose that the software in that product category is so substitutable, that all 100 members now using that category will stop buying it on the retail market and use this developer's product via the co-op. Let us ignore for now the potential capture of deadweight loss by some of the other 400 members, who may benefit from using this product even though their benefit is less than $50 so they weren't buying it retail.

Since bundling can make this kind of scenario play out in various software categories, all members should find some co-op funded software that they want to use. (Those who do not will exit the co-op.) If there are various products where a member pays $1 for software that only 20% of members use, then on average four times out of five the member is paying for software she doesn't use. So on average in the above type of scenario, a member who uses that software product is paying $5/year for it. The 90% savings off the $50 retail price comes from the fact that the developer doesn't care about his competitors' revenue losses.

By contrast, a software maker with a leading 90% market share might only be willing to give the co-op a 10% discount off retail price. Thus competition among bundlers to offer the co-op maximum benefit per cost, will tend to result in buying software from challengers in substitutable product categories. Note that this analysis does not depend on the co-op growing to large scale; a few hundred members would suffice. The next section below outlines further benefits attainable when the co-op grows to a larger scale.

7. Growth stages of the co-op

The co-op will require a significant amount of software infrastructure in order to launch. The voting system at votermedia.org is simple enough. The most substantial new component needed is the software user authorization system. It should be seamless to member users, continuously updated with the member list, and flexible enough for bundlers and their software suppliers to change their software offerings at any time.

8. There is a free open source substitute at OpenOffice.org.

The benefits to users and providers described in section 6 above should be enough to create a viable collective market. $60 per year per user can buy access to a range of low-priced software, so some providers will be attracted, even when there are only a few hundred members. The first software to be provided via the co-op may be from large developers of shareware, who can offer a ready-made bundle of their own products. Websites like ConsumerReports.org that publish respected information appealing to a wide range of users could likewise sell their own bundle. But gradually we can expect new bundler intermediaries to start up, who contract with various software providers for usage rights to a range of products, which they then sell to the co-op as a package. Competition for user votes will determine which software bundles are most valued. Some providers not satisfied with the revenue they receive from the co-op may stop supplying their software.

The co-op will learn from experience with this new market, and adjust system designs based on member feedback. Perhaps the $5/month price should be adjusted, or another collective funds pool added, for example at $20/month. Another reason for adding a second pool is if demand for different types of software comes from sufficiently distinct market segments. For example, there may be strong demand for game software from one subgroup of members, while most other members have no interest in it. The tools for helping members vote intelligently as described in section 5 above will be built and enhanced.

Bundlers will build their brand reputations, not only for supplying useful software, but also for reviewing the other bundlers' software, hosting member forums, advising the co-op on improving its organizational system designs, and perhaps contributing some infrastructure software to the co-op.

Members will have an incentive to recruit new members, because that will pay for more software. Even if the monthly fee per member stays at $5, competing sellers are likely to accept lower prices per member per software product as membership grows. More software for the same $5 price will in turn attract more new members. Lower prices per software product will also reduce the incentive for users to circumvent copy protection, so more will be willing to join the co-op and start paying for software.

If co-op membership grows to a significant fraction of the global retail software market, then it will start to affect market conditions. Smaller developers who started selling to the co-op early will grow larger, funded by their growing revenue. Some larger, more established developers may start selling via the co-op to maintain market share. The prospect of revenue from a large co-op will make it more economically feasible for developers to challenge larger competitors holding natural monopoly positions in their market niches. The "first mover advantage" will therefore diminish, as software markets become more contestable. The price of substitutable information goods will gradually be competed down closer to production cost, as the market becomes less monopolistic and more monopsonistic.

For non-substitutable software however, the co-op may be a boon to creators. Their revenue may increase because bundling helps them capture deadweight loss, as indicated by the analysis in Bakos & Brynjolfsson (1999).

The structure of software development firms may evolve in response to the growing opportunity to sell via the co-op. Mergers and vertical integration with a bundler may form a larger software firm, big enough to create its own bundle to sell directly to the co-op.

Competition to win the collective approval of a user community with paid expert advisers, may affect the features of software created and offered to a large co-op. Bundlers' long-term need to build and maintain their reputations may induce more subtle user-friendly behavior than we currently see in retail software markets. Hard-to-observe features like privacy and security may be enhanced. Developers may be rewarded for cooperating with each other more. Data may be more user-controlled, with easier export from and import to different makers' software systems.

Excludability is important for growing the co-op, to give prospective members an incentive to join and pay. But if it approaches global scale, then the co-op may start providing non-excludable benefits, i.e. to all software users, not only its members. Some of these may be spillover effects such as increasing competition on price and quality. Others could be the low-hanging fruit of non-excludable benefits whose benefit/cost ratio is so high that the cost is justified even if you only count the benefit to members. For example, co-op funded developers may make some of their software open source, especially if doing so enhances their reputations in the eyes of members. That would fulfill fundamental co-op principle #7: Working for the sustainable development of the community -- in this case, the global software community.

Another non-excludable benefit that co-op bundlers may provide is internet policy advocacy, like that of organizations such as openmedia.org and EFF.org.

8. From software to politics

The growth of a large international member-funded democratic organization would be an inherently political development. The software co-op's members would be voting funds to bundlers and to co-op staff, as a reward and encouragement for serving members' interests. Thus in effect the bundlers and staff would be elected representatives of a global internet community.

Legal and regulatory policies for managing the global internet (and more broadly, information technology) are complex and contentious. As with most government policy areas, there are many potential conflicts between the public interest and narrow private interests. Citizens may not be well informed, so private interests can often sway government decisions in directions harmful to the public interest. A global software co-op could provide internet users a funded organized expert voice to oppose such harm. Bundlers who provide that voice are likely to be rewarded with member voted funds. Governments (and their citizens) would know that these voices are backed by the voters in a global democracy.

The co-op may therefore become a public interest influence on such issues as internet privacy, free speech, security, spam, competition, and telecommunications. On intellectual "property" (IP) in particular, the co-op could not only fund advocacy for better law and policy, but also provide a model as a new way to reward innovators by voting them collective funds. This funding model may lessen the need for strong IP protection.

Without IP protection however, innovation becomes a non-excludable public good, and thus difficult to fund by voluntary payments like co-op member fees. Public goods are typically funded by compulsory payments -- taxes. But a software co-op can show how to allocate community funds (like taxes) more efficiently and democratically, using competitive voter-driven markets for public goods. This would close the loop back to where votermedia was first tested, in democracies.

Implementation in a software co-op would further validate the votermedia funding allocation mechanism, and encourage its spread to more democracies, as well as to corporations (for which it was originally designed). Unlike this paper's software co-op proposal, democracies and corporations can get considerable benefit from allocating as little as 1% or less of their budget by competitive vote. Competition channels those limited funds into whatever has the highest benefit/cost ratio, which tends to be advice to voters that will affect how elected leaders manage the other 99% of the budget. In a typical democracy, that role is served by investigative journalism, election coverage and public policy critiques. In a corporation, the main proposed service is proxy voting advice, but could also be expanded to include management strategy advice. This would increase accountability and reduce agency costs in democracies and corporations.

In spite of its potential benefits, votermedia has not yet spread through democracies and corporations, mainly (I think) because it would shift power from elected leaders to voters, so elected leaders tend not to implement it, while most voters are not yet aware of it. A successful software co-op could change that.

Appendix: A simple model of co-op voter decisions

Assume --

(a) The software co-op is structured as described in this paper, except that to simplify this analysis:
(i) members can vote for any allocation between 0% and 100% for each competitor (not just a few discrete menu options);
(ii) votes are therefore not interpolated between menu options;
(iii) the "spread" factor is omitted (or set = 1);
(iv) voting and award calculations are in continuous time.

(b) The co-op has grown large enough that the impact of a single vote can be analyzed as an infinitesimal increment using calculus first-order conditions, and other members' votes can be treated as a continuous density rather than discrete votes.

(c) All relevant functions are continuously differentiable.

(d) Each voter votes selfishly, to maximize the value of her vote's impact on her expected benefits.

For the vote aggregation rule described in section 3, the above simplifying assumptions imply that a vote's impact depends only on whether it is above or below the competitor's current award level. So each vote can be sufficiently described as either "for" a competitor (i.e. voting for more than the current level) or not. The option to specify a % allocation is merely a way of automating an on-off switch for the convenience of voters who don't want to revise their votes continuously. If a competitor's allocation rises above the % allocation you voted, then your vote for that competitor is switched off.

A given voter is trying to decide whether to vote for competitor c*, who is currently receiving some positive flow of funding from the co-op:

Define --

v = the current cutoff number of supporting votes required to receive funding

$share_c$ = funding share of competitor c

s_c = [slope] rate of decrease in $share_c$ per unit increase in v
= [for funded competitors] 1 / density of votes (rate of losing support) for c at current funding level
>= 0
= [for unfunded competitors] 0

Δv = [incremental] increase in cutoff v if this voter votes for competitor c*

$\Delta share_c$ = [for funded competitors] change in $share_c$ if this voter votes for competitor c*

Then --

$$\Delta share_{c^*} = (1 - \Delta v) s_{c^*}$$

$$\Delta share_{c \neq c^*} = (- \Delta v) s_c$$

$$\sum_{all\ c} \Delta share_c = 0 = (1 - \Delta v)\, s_{c^*} - \Delta v \left(\sum_{c \neq c^*} s_c \right)$$

$$\Rightarrow \Delta v = s_{c^*} / \left(\sum_{all\ c} s_c \right)$$

$$\Rightarrow \Delta share_{c^*} = s_{c^*} \left(\sum_{c \neq c^*} s_c \right) / \left(\sum_{all\ c} s_c \right)$$

$$\Delta share_{c \neq c^*} = - s_{c^*} s_c / \left(\sum_{all\ c} s_c \right)$$

Define b_c = marginal rate of increase of this voter's benefit, per unit increase in funding for competitor c.

Then the voter's incremental benefit from voting for competitor c* is --

$$\sum_{all\ c} b_c \Delta share_c = \left(b_{c^*} \sum_{c \neq c^*} s_c - \sum_{c \neq c^*} b_c s_c \right) \left(s_{c^*} / \left(\sum_{all\ c} s_c \right) \right)$$

which is > 0 if and only if --

$$b_{c^*} > \left(\sum_{c \neq c^*} b_c s_c \right) / \left(\sum_{c \neq c^*} s_c \right)$$

which holds if and only if --

$$\mathbf{b_{c^*} > \left(\sum_{all\ c} b_c s_c \right) / \left(\sum_{all\ c} s_c \right)}$$

Thus each voter will vote for the competitors with greater than average expected marginal benefit, where the average is scaled by the vote density at current share levels, and only competitors currently being funded are included in the average. This same rule applies to both funded and unfunded competitors.

References:

Bakos, Yannis and Erik Brynjolfsson (1999). "Bundling Information Goods: Pricing, Profits, and Efficiency," *Management Science*, Vol. 45, No. 12, December 1999, pp. 1613-1630. [Link to working paper version.]

Bakos, Yannis and Erik Brynjolfsson (2000). "Bundling and Competition on the Internet," *Marketing Science*, Vol. 19, No. 1, Winter 2000, pp. 63-82. [Link to working paper version.]

Brynjolfsson, Erik, and Xiaoquan (Michael) Zhang (2006). "Innovation Incentives for Information Goods," *Innovation Policy and the Economy*, Vol. 7, pp. 99-123. [Link to working paper version.]

Fisher, William W. III (2004). *Promises to Keep: Technology, Law, and the Future of Entertainment*. California: Stanford University Press.

Latham, Mark (2007). "Proxy Voting Brand Competition." *Journal of Investment Management*, 5(1), 79-90. Available at votermedia.org/publications.

Latham, Mark (2012a). "Experiments in Voter Funded Media." Working paper, available at votermedia.org/publications.

Latham, Mark (2012b). "We Want Our Co-ops Back." Working paper, available at votermedia.org/publications.

Lessig, Lawrence (2006). *Code version 2.0*. New York: Basic Books.