Measuring Constant Quality Industry Output Prices for Software Services

Michael Holdway
Bureau of Labor Statistics
Division of Industrial Prices and Price Indexes
Revised August 2003

The views expressed represent those of the author and not those of BLS or any of its staff.

I. Introduction

One of the oldest and most formidable challenges confronting statistical agencies is how best to disentangle price from quality change. The economic model and assumptions underlying a price index (such as input or output perspectives) provide theoretical guidance to the practitioner. However, theory often must yield to pragmatic approximations based on partial data sets and less than perfect knowledge and cooperation from survey respondents. As a result, quality valuation methodologies actually used in the operational environments of pricing agencies are often somewhat judgmental and require practitioners to develop extensive product and market knowledge.

This paper focuses on the problem of measuring both price and quality change for software services. More specifically, software services from an industry output perspective. The first several sections explore some general industry characteristics that are needed to satisfy the requirement for product and market knowledge. Then, starting in section VII, the paper takes on the immodest goal of assessing the practicality, relevance and transparency of price measurement and quality change valuation methodologies actually used or considered in the U.S. Producer Price Index (PPI) program. The assessments are deliberately pragmatic because contextual utility is best achieved when one does not lose sight of the need for accessibility in real time monthly production environments. The software industry is an especially interesting case because it reminds us of the lack of symmetry between the competitive assumptions of simple economic models and real world markets of dominant producers, network effects, and imperfect consumer knowledge. It is the real world market dynamics that contribute to uncertainty in the choice of the "best" practical measurement framework.

II. Software Services Broadly Defined

The U.S. National Income and Product Accounts (NIPAs) include three primary software services outputs that can be generically labeled as prepackaged, custom and own-account. Prepackaged and custom are industry defined outputs but own-account is described by the Bureau of Economic Analysis (BEA) as *in-house expenditures for new or significantly-enhanced software created by business enterprises or government units for their own use.*

The U.S. PPI introduced price indexes for the outputs of the prepackaged software industry in January 1998. Data, methodological and resource constraints make it unlikely that the PPI will introduce price indexes for custom or own-account software in the near future.[1] Establishments sampled by the PPI are primarily engaged in design, development, marketing, and production of system, application, utility, and entertainment software. As technology has advanced, the term prepackaged has expanded in use to include not only CDs or floppies contained in "shrink-wrapped" packages, but also software distributed electronically.

[1] Prepackaged software is now called Software Publishers under the North American Industrial Classification System (NAICS) and makes up NAICS industry code 5112. Custom software is now called Custom Computer Programming Services under NAICS and makes up NAICS product code 541511, which in turn is part of NAICS industry code 5415-Computer System Design and Related Services. The PPI is in the process of converting to a NAICS publication structure, which will be introduced for the durable, non-durable and service sectors in January 2004.

III. Industry Organization

According to data from the U.S. Census Bureau, the prepackaged software industry supported more than 12,000 establishments in 1997.[2] Though somewhat dated, the 1997 data reveal some interesting details about industry structure. Of the 12,000 prepackaged software producers, only 829 had revenues of $10 million or more. This last fact is important on many levels including sampling strategy, because while the 829 "large" establishments account for less than 7 percent of all establishments, they generate 78 percent of industry revenue ($48 billion). If we classify as small, those establishments with revenues less than $1 million, then small establishments account for 50 percent of all establishments, but only 2.4 percent of industry revenue.

The Census Bureau has also released interim data (for periods between the 1997 and 2002 full economic surveys) that enable year-to-year revenue comparisons at the industry and product class level.[3] Table 1 summarizes this annual data and shows that revenues for the prepackaged software industry (Software Publishers) jumped almost 50 percent from 1997 to 2001 and that custom software, which is approximately 2/3 the size of prepackaged, has grown about 65 percent.

Table 1. U.S. Prepackaged/Custom Software Revenues (billions of dollars)

NAICS 5112	2001***	2000	1999	1998	1997
Software Publishers	**90.62***	**88.04***	**80.95***	**72.09***	**61.70**
Personal computer software	17.91	13.81	12.94	11.81	n/a
Enterprise software	40.21	24.75	22.84	20.58	n/a
Mainframe software	10.37	8.70	8.67	8.06	n/a
Systems management software	N/a	13.80	12.26	10.13	n/a
Other services**	19.83	24.24	22.95	20.73	n/a
NAICS 541511					
Custom computer programming	**63.35**	**65.64**	**58.31**	**49.14**	**38.30**

**Revenue data for detailed software services do not sum to the aggregate for software publishers due to the withholding of data for certain software categories such as Electronic commerce enabling software.*
***The Other services category includes revenues received by software publishers for services such as user training, Internet access fees, web hosting/design and web site advertising.*
****The 2001 data is preliminary and no longer includes data for prepackaged systems management software. Because systems management software revenues are now included under the personal, enterprise and mainframe software categories, direct revenue comparisons between 2001software categories and prior periods are not possible.*

IV. Business Model

A. Industry Organization and Market Segmentation

Business practices and pricing strategies vary significantly among prepackaged software establishments due to proprietary technologies, diversity of product offerings, differences in

[2] The U.S. Census Bureau conducts a comprehensive economic census every 5 years that is used by the PPI as a guide for index structure as well as weights. The most recent comprehensive economic census was conducted in 2002 and is scheduled for release at the detailed industry level in 2004.
[3] The conversion from a SIC-based industry structure to NAICs changed the description of software industry outputs to an extent that precludes direct comparisons at the detailed level.

market share, market growth, installed base, and whether the primary target market is consumer or business, niche or mainstream. The complex and heterogeneous industry organization supports multiple and often contradictory analysis by experts. Part of the complexity stems from the need to establish standards critical for broad market acceptance of software services. Establishing standards is a form of network effect that often requires companies to use their intellectual property and patents to simultaneously compete and cooperate. Rather than get bogged down in a futile attempt to comprehensively detail industry idiosyncrasies, a broader outlook is more practical for the purpose of this review.

Primary prepackaged software services can be crudely but efficiently described as a synergistic three-level hierarchy. The first level is made up of programming languages and compilers used to create all the software that make up the other levels. Operating systems (OSs) make up the second level and provide a platform for the third level, which is composed of applications and games that provide utility directly to end-users. A few producers, including Microsoft, Sun Microsystems and Apple Computer develop and market prepackaged software across two or more levels of the hierarchy, but the majority of producers usually concentrate their efforts on one level.[4] The three companies mentioned all produce OSs but their markets are bounded to different degrees because an OS is usually designed for specific microprocessor architectures. Sun's Unix-based Solaris OS is designed to work on Sparc microprocessors, Apple's MAC OS-X for Power G4 microprocessors and Microsoft's Windows for Intel microprocessors.[5] Unlike Sun and Apple, Microsoft does not also build computers, which frees them to design OSs for the most dominant microprocessor in the computer industry (in terms of units shipped and revenues). In contrast, Apple has a 3 percent market share in desktop computers, which places a significant limit on the potential market for MAC OS-X. In order for Apple's OS to take market share from Microsoft, Apple must also compete with and take market share from producers of Intel-based desktop computers. Industry analysts have speculated several times in recent years that Apple would design their OS to work on Intel microprocessors as a means to expand their potential market and compete more effectively against Microsoft. As for the 3rd level of the hierarchy, the network effects of the dominant OS and microprocessor are a powerful incentive for software producers (including Microsoft) to develop applications and utilities that work with Microsoft OSs and Intel microprocessors.

The initial focus on OSs is due to growing definitional problems that blur distinctions between OSs, applications and utilities. If one were to take a pristine view, then the scope of OSs would be limited to software that provides master control functions for managing computer hardware and providing access points for other types of software to communicate with hardware and organize data files. The pristine OS definition could certainly be made more complex, but would still revolve around tasks that require managing input-output (I-O) requests from other non-OS software. However, the marketplace has evolved beyond the pristine view that was essentially in-place until the late 1980s. For competitive and consumer driven reasons, OS producers began

[4] IBM is an interesting case because they are a significant producer of proprietary OSs (AS 400 and AIX) and applications, but also play a major role in "open" platforms based on Windows or Linux and Intel. Gartner estimates that IBM mainframes with Linux OS accounted for 17% of their mainframe revenue in 2002.

[5] Sun recently introduced a version of its OS called Solaris X86 that is designed to run on Intel microprocessors. In addition, Sun is now marketing open-source Linux OS that runs on SPARC microprocessors. By offering Solaris X86 and Linux, Sun is trying to expand its potential market beyond their proprietary computer system applications and defend itself from market incursions by competitors.

to incorporate new features that went beyond basic I-O tasks to include web browsers, file compressors, disk defragmenters, music playback, games, movie editing and drawing/illustration programs. One of the consequences of a growing application-like feature set for OSs is that producers of stand-alone applications may find that OSs have co-opted a significant share of their market. In other words, the application provider's market may quickly change from one of competition with other application providers to one in which competition expands to include the OS provider. One of the concerns that application providers have with this type of competitive shift is that OS providers often add application-type features at no additional cost to consumers. Therefore, the application provider must develop products that are sufficiently superior and more desirable to what is included for "free" in the OS, if they expect to survive.[6] Alternatively, application providers can shift resources to the development of software services that have not yet or are unlikely to be included in future OS versions or in extreme cases appeal to the legal system for relief.

The market factors mentioned above were not intended to be comprehensive, but to suggest that rapid industry evolution not only brings new services to market but also can significantly alter competitive relationships in ways that are difficult to predict. New services, dynamic competitive landscapes and establishments with significant non-software outputs (such as computers) that account for large intra-company transfers make a statistical agency's sampling and repricing strategy more complex.

Before exploring some of the industry specific sampling and repricing issues it may be prudent to first review the basic terms of transactions that account for the majority of industry revenue. As a starting point, prepackaged software is not "sold" in a traditional sense; rather producers license the right to use their software. Right-to-use terms can be perpetual or limited to specified time periods. The latter is growing in importance as producers experiment with various subscription or annuity pricing models to obtain more predictable revenue streams. Generally the usage terms set forth in a license do not allow or restrict the consumer's ability to resell, transfer, reverse-engineer or alter, though licenses for open-source software can be less restrictive.[7]

Commercial licenses account for the majority of industry revenue, but commercial and consumer licenses have similarities in that they generally disclaim all liabilities for anything that may happen to a user's hardware and software when the newly installed software is running. The relatively narrow view of liability implicitly acknowledges that even large software firms cannot pretest for compatibility with every possible computer configuration nor can they guarantee that "bugs" are not present in programs that routinely exceed millions of lines of code.[8]

---

[6] Superior technology may not be a guarantor of survival. There are many examples of products that were technically superior to competitive alternatives that failed. If consumer choice is expanded to include what is free and "good enough" vs what is not free but more feature rich, then consumers have demonstrated that "free" is a compelling choice.

[7] The terms of use for popular open source software such as Linux or Apache are provided under a General Public License (GPL) that is frequently distributed at no or nominal cost. GPLs give users the right to change the source code as long as they make the changes freely and publicly available. Open-source software began to penetrate the corporate market in the late 1990s and today includes desktop and server operating systems, desktop applications suites (Open Office), web server software (Apache) and databases (MySQL and Red Hat Database).

[8] According to a study funded by the U.S. National Institute of Standards and Technology (NIST), software *bugs and glitches cost the U.S. economy about $59.5 billion a year.*

If a statistical agency's sample of transaction prices were limited to software license fees then the universe of industry revenues has been significantly truncated. The composition and growth of industry revenues is increasingly dependent on maintenance contracts with commercial and government accounts.[9] Maintenance agreements are usually tied or bundled with software licenses and based on fixed terms of 2-4 years. The maintenance fee is a annual payment (typically 20-30 percent of the original software license fee) that give buyers the right to upgrade at no additional cost to subsequent software versions. Maintenance fees are essentially pre-payments for future versions of software and place additional pressure on software vendors to introduce new versions (if new versions are not introduced then the maintenance fee is not refunded). Buyers that have a history of frequent upgrades to new versions are probably the least resistant to the prepayment model. On the other hand, many buyers prefer not to upgrade whenever a new version comes to market. Microsoft estimates that it has a 240 million user installed base for its Office software suite, but 60 percent of these users are still on Office 97 or an older version. If the users of Office 97 or older could have been convinced to pay for upgrading to the new version, Office XP, in 2001 then Microsoft would have gained an additional $10 billion of revenue (DeGroot 2001). So it is clear that one of the important challenges facing software producers is to entice their installed base of users to upgrade to the most recent version. In addition to new features and bug fixes, some producers try to entice users to upgrade with other incentives. For example, as a software version ages, the discounted price of an upgrade to a new version increases. Upgrade fees that are not part of maintenance agreements are generally between 40 to 70 percent of the original license price and as installed versions age the upgrade fee scales to the higher end of the range. Producers may also decide to discontinue all support for an older version, which is another way to encourage users to upgrade.

The issue of upgrades has grown in importance as the software industry matures. Many producers have encountered slowing growth partly because of long-established, richly featured products and deep market penetration. In order to grow revenues, producers continue to target customers of competitors but increasingly work to convince their own customers to upgrade. Upgrade resistance appears to be the primary motivation behind experiments with new license terms and other sales inducements. However, anecdotal evidence seems to indicate that, at least so far, a majority of users continue to resist more frequent upgrades and in fact are moving in the opposite direction. Tom Dubois, an analyst for ARS research, observed that *…it's getting harder and harder to convince consumers and businesses that they need to upgrade…Changes in the last few versions of Office have been incremental, and most folks already have all the word processing capability that they need*. Dubois goes on to say that, *It's no secret that the advantages of upgrading operating systems or application software have diminished quite significantly over the last few years. If you look back over history there were great advantages from one release to another. You just don't get that anymore.*[10] As previously mentioned, the primary motivation for producers to push for continuous upgrades is to obtain a more predictable and steady revenue stream. ZDNET.com reported that at Microsoft's 5th annual CEO Summit, Bill Gates mentioned that software was becoming a tough sell and went on to say that, *Intellectual property has an interesting problem, which is that it lasts forever. Your own*

[9] Microsoft began offering what they call the Software Assurance Upgrade program to their volume-licensing customers in 2002. Software Assurance replaced a more lenient upgrade program and is paid as an annual fee (approx. 29% of the software license fee) over renewable two or three year terms.

[10] Excerpt from CNET News article, May 29, 2003, "Microsoft rethinking its Office plans" authored by David Becker.

installed base is serious competition. If producers are successful in replacing perpetual with subscription licenses then the industry problem of old software versions effectively competing with current versions is reduced.[11]

Bill Gate's comment on the market effects of intellectual property longevity offers a transition point to some final observations regarding industry characteristics. Software has the interesting property that as the cost of developing intellectual property that is transformed into programming code are recovered, the marginal cost of producing additional units of software drops tremendously. Estimates for the cost of reproducing software code on CDs are about 50 cents per disk.[12] As broadband connections become more prevalent, the cost of distributing software electronically reduces marginal cost almost to zero. It is the upfront costs of developing new or new versions of software that play a central role in determining pricing strategy and volume requirements or even whether a project should proceed. The development activities are perhaps most simply described as resources directed to the design and/or rearrangement of strings of algorithms that may offer hundreds or thousands of features embedded in millions of lines of code.[13] Producers often say they compete more on features than price, but clearly the development of new or improved features is one of their most resource intense and price determining activities. The substantial sunk costs represented by software development will to a large extent determine pricing strategy and which market segments are targeted[14] An expanded discussion of sunk costs as they relate to new versions of software is deferred to section VIII.

B. Government Regulation

Government, at various levels, has significant interaction with the software industry. As previously mentioned reproducing the intellectual property embedded in software has a low marginal cost.[15] This low reproduction cost can be both a blessing and a curse for the software industry. One of the negative consequences is that it is relatively easy for unauthorized individuals or organizations to make copies of prepackaged software for resale or consumption. IDC recently completed a study (see references) for the Business Software Alliance that investigated the incidence of software piracy in 57 countries. IDC concluded that; *globally, four out of ten copies of software are pirated, with piracy rates in individual countries ranging from 25 percent to 94 percent.* Despite the rampant theft of intellectual property, IDC also estimated that *spending on software grew six times faster than spending on computer hardware between*

[11] Subscriptions should reduce the initial software acquisition cost for users (relative to perpetual licenses), but then require periodic (usually annual) fees to maintain right to use. If subscription fees are not paid, then the software must be uninstalled. Subscription plans, as currently structured, normally entitle users to future upgrades at no additional cost so long as the periodic subscription fee is paid.

[12] Reproducing software media is a part of the prepackaged software industry's output, but is a specialized process often outsourced by software developers. The Census Bureau estimates the value of purchased software reproduction grew from $1.9 billion in 1998 to $2.5 billion in 2001.

[13] Microsoft's latest operating system, Windows XP, contains more than 47 million lines of code, while its predecessor, Windows 2000, contains about 20 million lines of code.

[14] In a Mike Ricciuti interview for ZDNET.com on 6-20-01, Bill Gates said …*Microsoft has always been extremely focused on high volume, low price, we're not interested in things that we only sell to hundreds of thousands of people. So we have to come up with a value proposition and simplicity that makes this attractive to millions and millions of people.*

[15] David Coursey, executive editor of Zdnet's Anchor Desk recently stated that, *One of the great things about software is that while creating the first copy is very expensive, the second and all subsequent copies are free. Sure, you can spread development costs over every copy distributed, but the money was actually spent only to produce the first one.*

1996 and 2001. Governments benefit from this rapid growth due to increased tax revenues.[16] But the actual growth in tax revenues may be substantially lower than the potential growth due to high piracy rates. Therefore curbing piracy not only benefits the software industry, but also governments with significant IT interests. Partly as a result of this dual potential benefit, the governments of countries with large IT sectors have become more aggressive in tying protection of intellectual property to bi-lateral and multi-lateral trade agreements.[17]

Another interesting and controversial government-industry interaction is a growing tendency for some governments to openly promote increased internal use of open-source software. Australia, China, Japan and Korea have been especially active in expressing strong interest in the Linux open source operating system. According to a Reuters report, the city of Munich recently decided to convert 14,000 computers from Windows to Linux. As for the U.S., Linux is used in many government agencies, including the Department of Defense. A recent National Aeronautics and Space Agency (NASA) research paper (Moran 2003) concluded that open-source software would be appropriate for and benefit many NASA projects. The proponents of increased open-source use in government are not always focused entirely on potential cost savings. Instead, the ability to evaluate security vulnerabilities more completely due to unfettered access to source-code is a significant consideration. Commercial software producers tend to view source-code as part of their "crown-jewels" and have generally been reluctant to make this code available to users. However, growing security concerns have caused many government agencies to consider the greater transparency of open-source as a significant advantage. Partly in response, Microsoft has reversed previous practice and is revealing Windows source code as part of its Government Security Program (GSM).

Governments, especially in OECD countries, also monitor industry and company practices that may act to restrain trade or competition. In the high visibility *US vs Microsoft* legal proceedings, arguments became highly technical tugs-of-war in defining what is an operating system and whether bundling additional features in the dominate OS constituted an unfair trade practice. The initial judgment in the Microsoft antitrust case required the company to be split in two, but this remedy was overturned on appeal as too severe. Microsoft's conduct was still determined to be illegal and several preliminary remedies have been put in place. The remedies are designed to stop business practices that were found to help Microsoft maintain a monopoly position in desktop operating systems. In Europe, a review is currently underway that examines Microsoft's position and marketing practices in server and desktop operating systems. The point here is not whether these legal proceedings are good or bad for the industry or consumers, but that as the software industry grows relative to GDP, it is likely that government interaction with the industry will grow on multiple fronts. As a consequence, it is possible (some might say likely) that anticipation of potential regulatory response will more obviously influence future marketing and pricing practices of large software companies.

[16] Not surprisingly, the rate of software piracy drops as the size of the IT sector grows as a percentage of GDP.
[17] The IDC report also contains multiple examples of reduced piracy rates on a country-by-country basis that shows an average 10-point reduction for 37 of 57 countries since 1996. IDC estimates that each 1-point reduction in software piracy, adds $6 billion (world-wide) to tax revenues.

V. Sample Design

A. Sample Frames

At an aggregate and conceptual level the universe for the U.S. PPI are all domestic producer transactions for all industries in the U.S. economy, excluding imports. Because a comprehensive and accurate listing of all transactions is not feasible, the frame sources used by the PPI are an approximation. At the industry level, the frame source used for PPI samples is usually the Unemployment Insurance Files (UI) records reported to states for the purpose of assessing unemployment insurance taxes. UI files are generally the most comprehensive and cost-effective frame source available. However, UI files are not without problems, particularly for clearly identifying what business operation the UI file is describing. To filter and correct the most obvious identity and classification problems, PPI industry analysts use a frame refinement process that requires direct contact with the largest frame units (establishments) to verify measures of size (employment), primary output and data records center location. The refined frame is then used to select sample units based on a probability proportionate to size technique. The number of sample units selected to represent a specific industry's outputs are based on numerous factors such as industry size (dollar value of shipments), degree of producer concentration, publication goals at the detailed product level, available resources and if previously sampled, the historic price variance within and between establishments as well as attrition rates.

Designing and selecting industry samples for the Services sector often introduce additional challenges relative to goods-producing industries. For instance, the size of potential frames for many service industries dwarf the frames used for most durable and non-durable industries. Because of the previously mentioned definitional ambiguities in UI files, PPI resources (staff hours) may not be sufficient to effectively refine and correct the much larger service industry frames. Therefore, if alternative frames, often from a commercial source, are available for large industries they may be evaluated as possible replacements or supplements for UI data.

The initial UI file for prepackaged software was compared to a purchased alternative frame. A large discrepancy was noted in that the UI file had records for 9,907 establishments, while the alternative frame listed 15,526 establishments. Further review showed that most of the additional establishments listed in the alternative had five or fewer employees. Fifty of the five employees or less establishments were randomly contacted as part of a pre-refinement test and it was determined that half of the establishments were out-of-scope (misclassified). Random crosschecks showed that the frames appeared to be similar for mid to large establishments. The PPI decided to use the UI file as the primary frame source partly because of more precise measures of establishment size as opposed to ranges of size in the alternative frame. However, to ensure the most representative sample, the alternative frame and PC Magazine were used in addition to the UI for purposes of identifying and then contacting the top 70 companies to confirm industry classification and records keeping addresses.

The previous description of industry organization noted that approximately 78 percent of the prepackaged software industry's revenues were accounted for by only 7 percent of establishments. Due to this high concentration, the PPI developed a sample strategy based on

explicit stratification by size. In other words, to increase the efficiency of a random sample, the refined UI frame was stratified into two groups based on employment size.

VI. Publication Structure and Relationship to CPC

The PPI's industry output indexes are currently structured around the Standard Industrial Classification (SIC) system. The detailed publication structure used by the PPI for prepackaged software was based on collaboration with the Census Bureau and the Software Publishers Association (SPA) and is shown in table 2. Disclosure concerns required suppression of publication for certain software categories and broad definitions for those software categories that could be published.[18] The Census Bureau provided weight (revenue) information at the 5-digit level, but at the more detailed 7-digit level, SPA recommendations for product descriptions and relative importance were used.

Table 2. PPI Publication Structure for Prepackaged Software

SIC	Industry and product
7372	Prepackaged software
7372P	Primary services
73722	Applications software
7372201	Applications software sold separately (non-suite)
7372202	Applications software sold as a suite
73723	Computer games and other prepackaged software
73724	Maintenance, documentation, training and other software services
7372SM	Other (non-primary) receipts

The Central Product Classification (CPC) adopted by many OECD countries does not provide a detailed prepackaged software service structure that would enable comparison with the PPI's SIC-based structure. However, the International Standard Industrial Classification (ISIC) Rev 3.1 includes group code 722-Software publishing, consultancy and supply that is an aggregate for class code 7221-Software publishing.[19] Explanatory notes accompanying class code 7221 describes in-scope output as *production, supply and documentation of ready-made (non-customized) software*. Examples given are:

- Operating systems
- Business and other applications
- Computer games for all platforms

[18] Price indexes calculated, but not published, feed into the Primary services (7372P) aggregate.
[19] I could not find a concordance between ISIC class code 7221 and the CPC.

VII. The Conceptual Price Measurement Framework and Real World Challenges

A. Background

The U.S. PPI takes a conditional industry output point of view restricted by a fixed production function established by the reference period's output. At a macro level, a fixed production function implies that the growth rate of an output price index should be proportional to the growth rate of industry revenues. A Laspeyres perspective is used by the PPI to approximate this measurement framework called the Fixed-Input Output Price Index (FIOPI) model. The FIOPI theoretically enables industry revenues (outputs) to be compared on a constant quality basis so long as input requirements and technology are held fixed. Of course the PPI cannot directly measure the universe of an industry's transactions that make up revenue, nor can it precisely define and quantify all of the input requirements that establish an industry production function. Therefore, the macro measurement is approximated at the micro level with a probability proportionate to size sample of specified transactions for specified products that represent industry revenue and for which input requirements are likely to be more transparent. As long as input requirements are static, then statistical agencies can publish adequate industry output price indexes with good sample designs and "matched models". When survey respondents report changes in the characteristics of sampled products or services, they are, in effect, signaling that input requirements might have changed, which violates the fixed inputs assumption of the FIOPI model. In other words, real world dynamics have collided with the tidy, but static assumptions of the FIOPI. How pricing agencies respond to these collisions between the real and theoretical worlds is often of keen interest to index users. It is clear that the more dynamic an industry's output, the more quickly the default "matched model" breaks down.[20] In order to maintain index continuity pricing agencies normally replace discontinued products with replacements that may have new or additional features enabled by new inputs. The challenge for the pricing agency then becomes how to account for new input requirements when comparing the price of the replacement with that of the predecessor. Price indexes based on the FIOPI generally default to the resource cost approach when valuing changes in output quality. Valuing quality change by the marginal cost of new input requirements is, in theory, a methodology that helps disentangle price change from shifts in a production frontier.[21] Fisher and Shell (1972) developed many of the theoretical details that support the resource cost method for constructing constant quality industry output price indexes.

B. Price Measures for Prepackaged Software

The prices reported voluntarily to the PPI by prepackaged software producers are primarily based on transactions for single and multi-user licenses sold to computer OEMs, distributors, retailers, large business accounts and in some cases direct to the consumer. The terms of commercial software licenses must be clearly specified because transaction prices in the commercial market are most often based on how software is actually used. For instance, the same software may be licensed on a per seat (user), per device, per processor or on a concurrent

[20] Ignoring for the time being the potential issue for out of sample new item bias.

[21] More precisely, the marginal costs directly associated with quality change plus the producer's standard markup is the default quality change valuation used in the PPI.

access basis each with a different pricing structure.[22] Per user and per device fees are the most straightforward price measures and normally used for desktop OSs and applications. Per processor or concurrent licensing terms add complexity and are most often encountered for OSs and applications designed for computer servers.

An example of a server OS or application license fee might be $10,000 if installed on a 2-cpu server, but increase to $15,000 if the identical OS or application is installed on a 4-cpu server. In effect, the price of software sold under per processor terms is scaled according to use, with greater customer utility (speed of processing) priced into the 4-cpu software license relative to a 2-cpu deployment. Alternatively, software designed for servers may be licensed on a fixed fee basis per server similar to personal computer software.

Complicating price measurement of server or client software is an additional fee, sometimes referred to as a Client Access License (CAL). The license terms for server software may allow installation but not access. Access to the software on a server in some cases is only allowed through the purchase of CALs for each client that might connect to a server. One of the interesting features about CALs is that unlike the other licenses discussed so far, CALs do not directly include prepackaged software. CALs simply confer the right to each client to access software that has been installed on a server. Continuing with the previous example, if CALs are priced at $25 per client then the total license fee (software + CAL) for 300 clients, to access an application installed on a 2-cpu server is:

Software Application License Fee for 2-cpu server	$10,000	
CAL for 300 Clients	$ 7,500	($25 CAL X 300 Clients)
Total License Fee	$17,500	

There are many potential CAL implementations that if unaccounted for could negatively affect the deflation properties of a price index based only on the more visible direct licenses. CALs also raise an issue for econometricians interested in building hedonic models to construct constant quality price indexes from secondary source data. Software producers can and do use CALs as an inducement for customers to upgrade old versions of desktop software. For instance, if a customer has the most current office suite or other applications installed on clients, then the CAL fee for new server applications is often waived. In other words, the cost of older software versions that co-exist in the market with new versions is effectively increased relative to new versions by the amount of the CAL. If the dependent variable in a hedonic model does not reflect this additional transaction cost over time, then the model becomes further disconnected from the real world market.

The type of price sampled is also important because many producers publish a list price that may be discounted 20-60 percent depending on the type of software, type of buyer, type of license and volume. Discounts or price adjustments for specified transactions, rather than list prices are

[22] Concurrent access is most often used for applications installed on computer servers. The license fee is based on the maximum number of users that may attempt to concurrently access the application across a network from a client device (desktop, notebook or even a PDA). Therefore, the more users accessing the server application, the higher the total fee charged by the software publisher for the application.

the most common price reported to the PPI. Both full and upgrade version license prices are also currently represented.

C. Price Measures for Open Source Prepackaged Software

A discussion of software pricing would be incomplete without reference to the rapidly growing market for open-source software. Contrary to popular perception, open source is not necessarily synonymous with free. Definitions of open-source are varied, but most commonly used to indicate that, at a minimum, there are no restrictions on customer access to source code. There are a variety of classes of open-source licenses that co-exist in the marketplace. One of the most common is called the GNU General Public License (GPL). GPL requires that any modification to source code covered by GPL must abide by GPL rules and made freely available to the public. GPL terms may be viewed incorrectly by some as incompatible with commercially motivated business models. A less restrictive alternative to GPL is the Mozilla Public License (MPL) that allows developers to combine their own independently created code with MPL code, but the hybrid code is not required to be made freely available to the public. As long as the MPL code is not changed, then MPL provides a more commercial friendly environment relative to GPL. One of the least restrictive open source licenses is the Berkeley System Distribution (BSD) that only requires that copyright holders be referenced in all accompanying documentation. BSD is considered the most commercial friendly of the common open-source licenses.[23] A BSD UNIX derivative serves as the core of Apple's new Mac OS X operating system software.

In terms of units physically shipped or distributed electronically, the Linux OS, which is actually a Unix derivative, is perhaps the most important open-source example. Several Linux vendors have gained prominence in the last few years as the OS became popular with Internet Service Providers (ISPs). According to IDC, various iterations of Linux accounted for nearly 26 percent of the server OS market (more than 50 percent for web servers) at the end of 2001.[24] The IDC estimates for Linux are startling because they are not based on units, but revenues. The Linux licensing model (GPL) does not preclude vendors from charging fees for services. Rapid revenue growth for open-source implied by IDC may be due in part to the adoption of business practices that are similar to those used for proprietary commercial software. Linux vendors add value to the OS by developing or customizing applications and utilities, which they bundle with Linux.[25] Fees are derived from licensing the software bundles and providing technical support and training. For example, one of the major Linux vendors sells enterprise versions of the OS for servers in standard and premium additions. The standard edition is offered for $1499 and the premium edition for $2499. In both cases the fees are subscription based and must be renewed annually. Highlights of the features and differences for the standard and premium editions that allows for price differentiation are listed below:

[23] There are many other types of open-source licenses and the descriptions provided for those listed above are highly simplified.
[24] IDC also estimates that in 2002 Linux software was installed on 15 percent of all servers in Western Europe.
[25] Some of the value-added may be as basic as simplifying installation for non-technical users or improving the appearance of the user interface.

Table 3. Linux OS Features for Servers

Standard Edition ($1499)	Premium Edition ($2499)
Handles up to 2 cpus and 4GB of memory	Handles up to 8 cpus and 16GB of memory
Installation support	Supports clustering
Configuration support	24 X 7 emergency support
Systems administration support	Installation support
Free upgrades within subscription term	Configuration support
	Systems administration support
	Free upgrades within subscription term

The features listed in table 3 are not comprehensive, but used to show that the vendor's pricing strategy for the premium edition is based on providing both increased software functionality and higher levels of support. The premium edition's capability of addressing up to 8 cpus found in high performance servers is an important feature for large enterprise class clients. The ability to make use of up 16GB of memory is a critical feature for hosting large database applications. While the level of service and support appear roughly similar between standard and premium, the addition of 24 X 7 emergency response in the premium edition is a feature demanded in "mission critical" environments (such as transaction processing). In many respects, the feature lists in table 3 are similar to or identical to what one might expect in the marketing materials of proprietary commercial software producers. The main point to be made is that open source software marketed to businesses is increasingly following an economic model that lends itself to measurement by statistical agencies.

D. Industry Marketing Practices that Impact Price Measurement

A few examples, by no means comprehensive, of industry pricing strategies that statistical agencies will encounter are listed below:

•Producers may offer "free" software applications, but include advertisements from 3rd parties embedded in the application. Alternatively, the same producer charges the customer a license fee to obtain an ad-free version of the application. In the first case, the producer "price" is the fee charged for 3rd party ad-placements and in the latter the price is a more typical end-user license fee. Due to changes in advertisement placement and potential audience (number of users), prices based on this model may be more volatile than prices based on end-user license fees.

•Producers may keep license fees static, but change the level of client support bundled with the software license. For instance, some producers have started to restrict the number of "incident" or non-installation related customer support inquiries to one or two and then charge $30-$40 for each additional support call.[26] If previous license fees included support for a specified time period, say six months, then the actual level of support may have changed if new license terms limit support to a maximum number of incidents. This change may be viewed as a price increase (all else equal) if on average the producer receives additional support revenues due entirely to

[26] Another method used by producers to enhance revenue (increase prices) is to change from a toll-free telephone support policy to an explicit charge based on length of call, such as .99 cents per minute.

changing from unlimited support for a specified time period to a fee structure for all support provided beyond a specified number of incidents.[27]

•Certain types of software outputs cannot be priced in the PPI. One of the most common examples is the growth of what are often referred to as service packs (SPs). SPs are new software code modules developed by the producer to fix "bugs" or security vulnerabilities in previously shipped software versions.[28] SPs may also add new features and functionality that are most often distributed as free electronic downloads or made available on CD for a nominal shipping and handling charge. To the extent that an explicit quality adjustment methodology is developed, then SPs incorporated into new versions of software could in principle be accounted for.

VIII. Technical Concerns

A. Quality Adjustment in Theory and Practice

The estimation of constant quality output prices using standard PPI methodology is dependent on a two-part process that is conceptually simple but often difficult in practice. First, the new input requirements associated with a feature or function change is identified and then the marginal cost directly associated with new input requirements are used to quality adjust nominal price relatives. As Triplett and others have noted, output indexes in theory should *measure the ratio of* (maximum) *revenues associated with remaining on the same production possibility curve in two or more periods.* When the PPI uses the marginal cost of new input requirements to value quality change, it is attempting to maintain the reference period's production possibility frontier in the comparison period so that shifts in the production function are not measured as price change, which requires that inputs and technology are held constant. When quality improves, the marginal costs associated with new input requirements establish the value of quality adjustment (VQA) in a price ratio that can take the form: $\frac{\mathbf{P_c - VQA}}{\mathbf{P_r}}$. When quality declines, the price ratio takes the form: $\frac{P_c - (-VQA)}{P_r}$; where $\mathbf{P_c}$ is the comparison period price and $\mathbf{P_r}$ is the reference period price. This is the default methodology (ignoring weight and aggregation issues) used in the PPI to approximate a constant quality industry output price index.

Precise measures of changes to input requirements for technically complex products are often not available from producers. When producers cannot provide quality valuation data, then absent an alternative explicit methodology, options are limited. If producers tell us or it is otherwise obvious that quality differences are significant, then the PPI often employs a implicit quality valuation methodology called direct link or link-to-show-no-change. Direct link treats the entire price difference between a new product and its predecessor as the value of quality change. In effect, the price difference between the new and old product substitutes for the marginal cost of

[27] A major producer recently announced a change in support policy that goes in the opposite direction. To encourage customers to stick with a somewhat controversial licensing program, support and training (previously provided on a fee basis) are now included for free.

[28] Many IT managers do not consider software reliable enough for large installations until the first service pack is released.

new inputs (VQA) in the price relative formulas shown above.[29] In the real world, the price difference between an old and new product may, in fact, be due entirely to quality change. If so, then direct link is by happenstance an unbiased price measure. However, if a real price change occurs for replacement products that incorporate new features or technology, then direct link cannot measure this change introducing bias of unknown direction and magnitude.

An alternative to direct link is an imputation based on price change for other similar products in the index. For instance, if an obsolete product is replaced with a new improved version and prices of other sampled competing products respond quickly to the introduction, then the observed price change for these other products can be used to proxy real price change at the point of introduction for the new product. Imputation works best in competitive markets where relevant information is available with low search cost that enable consumers to easily assess the price/utility of competing quality-differentiated products. With the "right" market conditions, one could reasonably expect that prices of older lower quality products would immediately respond to the introduction of new superior products to yield roughly similar price/utilities. The PPI experience is that prices for old versions of software such as an OS or Office suite often do not respond to the introduction of new versions. In fact, a common scenario is that new versions are priced identically to predecessors. When old versions coexist in the market at the same price as new improved versions, then direct link or imputation yield identical and upward biased results.[30]

B. Alternative Explicit Quality Valuation Methodologies

Practitioners (and many users) generally consider explicit quality valuations to be superior to implicit methods especially when products and markets evolve rapidly. However, in the absence of producer provided cost data, explicit methods require more resources relative to implicit. The McKinsey Global Institute (MGI) recently published a review of productivity growth trends in the U.S. As part of this review, MGI examined three methodologies that could, conceptually, offer explicit quality change valuations for software services.[31] The three methods can be generally described as:

- Hedonics
- Lines of code
- Function points

[29] If a new product is simultaneously improved in terms of quality and priced lower than its predecessor, then absent explicit data for quality valuation, prices will be directly compared. While still upward biased, direct comparison in this situation reduced bias relative to direct link.

[30] When prices of older competitive versions do not respond to the introduction of a similarly priced new version, then both direct link and imputation yield a VQA of zero or a price relative of 1.0. The PPI has also encountered situations where the old version price is unchanged, but the new version is priced higher. In this case, both imputation and direct link continue to yield a price relative of 1.0 at introduction. Only when prices of old competing versions respond directly to the introduction of a new version can price relatives from imputation and direct link diverge. There are many reasons for lack of price responsiveness from older competing versions that include imperfect consumer knowledge, additional costs for adopting new versions and producer marketing strategies that may differ from what might be expected under more competitive non-proprietary conditions.

[31] The MGI report received technical assistance from advisory committee members Robert Solow, Barry Bosworth, Ted Hall and Jack Triplett.

1. Hedonics

In the academic and statistical agency communities, hedonics is probably the most often-discussed quality change measurement tool. Hedonics is simple in concept, but has thus far been used in official price statistics primarily as a tool for measuring constant quality prices for computers and related equipment. The U.S. PPI developed hedonic models for computers on an operational basis in 1990 but has since developed models for only a few other products.[32] Due to rapid changes in technology (new characteristics), computer models must be updated on a regular basis (at least 2 to 3 times annually). The need to frequently update models after they are first developed represents a significant resource commitment for statistical agencies and may partly explain the limited role of hedonics in official statistics.

The MGI study rejected hedonics for valuing software quality change due to problems with model specification and lack of relevant data. Unlike models developed for computers, there are not hundreds of observations available for differentiated products that could in theory support a robust software model. Producer concentration in important software categories such as OSs and productivity suites do not support the large data sets required to reliably generate coefficient values for complex software characteristics. A related problem is that even if large data sets were available, what specific characteristics (out of potentially thousands for a single software service) are the most important indicators of resources absorbed and utility consumed.[33] Triplett (1986) notes that, *determining the characteristics of a particular product require a great deal of technical information, an understanding of what is produced as well as how it is used. It has not always been easy to assemble the technical knowledge. Nevertheless, good design of a hedonic investigation requires that the choice of variables be based on technical considerations about the production and use of the product under investigation.* This last issue is perhaps the most daunting, because network effects and other interactions such as between the OS and specific hardware technologies or between one type of software that must work with other types of software may be as or more important than a simple vector of embedded software characteristics.[34] The challenge of constructing a properly specified software model is unlikely to be resolved without much improved data transparency, more sophisticated understanding of how software is produced and a consistently defined and applied utility metric. Some may feel this last point borders on overstatement or creates hurdles that are not explicitly placed on other models used in official price statistics. However, as previously mentioned software has some rather unique properties in that it is intellectual property expressed by various combinations and re-combinations of algorithms that have become increasingly complex. It is the complexity of current generation software that makes any assumption about rough equilibrium between resources used and utility delivered, as implied in most hedonic functions, problematic.

[32] Hedonic models developed by the PPI are not used to directly construct price indexes as is often presented in research papers. Instead, the PPI uses hedonics to estimate values for changes in specific product characteristics that are directly reported by survey respondents. PPI samples are generally of sufficient size to represent industry output, but not large enough to support a richly specified hedonic model. Therefore, models are generally built from secondary source data, rather than directly from a PPI sample.

[33] This assumes that a data source is available that accurately describes characteristics sufficient to differentiate one product from another with meaningful market driven granularity. Otherwise one may fall prey to mis-specified models that generate characteristic and/or time dummy values of questionable real world relevance.

[34] According to Apple, their new operating system, Mac OS X v10.2, has more than 140 new features, such as faster graphics, better access for the physically challenged and an improved spam filter for e-mail.

Prepackaged software is still a relatively young industry but buyers, particularly at the corporate level, have begun to move up the learning curve in recent years due to extensive experience with the costs, both direct and indirect, of frequent upgrade cycles. Promises of improved productivity have in many cases only been partly realized as the problem of how to adapt existing business processes to software features rather than the other way around is beginning to play a more decisive role. Some of the salient contributors to this costly learning curve have been widely reported and may indicate problems with converting software outputs to productive inputs. Reports from various trade journals are summarized below that help describe the real-world technology absorbing predicaments faced by many business consumers.

- ZDNET News reported in 2002 that; *Many in business and government warn that large sums of money are often wasted when technology is thrown at problems that require organizational and cultural change. This is a lesson that businesses have learned over and over again in their effort to become more efficient and customer-friendly by implementing business applications from companies such as SAP, PeopleSoft, Oracle, i2 Technologies and other big software companies. Untold number of companies including Nike and Hershey Foods, have lost millions of dollars in failed software projects.*

- PC Magazine reported in "CRM on a Shoestring" (08-02) that; *companies have overestimated the adaptability of employees when introducing complex software that requires changes in a business process. Morgan Stanley recently reported that U.S. companies spent $130 billion more on software over the past two years than they should have. In addition to paying for unoccupied seats companies are also paying for software modules that are never used.*

- Gartner Group has estimated *the cost of converting a user from Microsoft Office to Star Office at about $1,200 including retraining and lost productivity during retraining*. If the Gartner estimate is in the ballpark, then the switchover expense may play a greater role in a purchase decision than a licensing fee or product features, which again places additional pressure on the researcher to specify a hedonic model that reflects relative real world transaction costs.[35]

- eWeek Labs conducted a survey of corporate IT departments one year after the Windows 2000 OS was released. The survey asked IT managers that were on previous versions of Windows OS why they had not upgraded to the new version. The reasons given were all based on post-purchase concerns included "enormous training requirements, network redesign, administration changes, application reconfiguration and mandatory upgrades of other software for compatibility.

- Directions on Microsoft, a newsletter devoted to all things Microsoft, reported that installation of the new Windows Server 2003 OS will require an upgrade to Exchange and that *Most earlier Microsoft server products also will not be compatible, and some third-party products will require patches or upgrades.*

- According to J.D. Edwards, a producer of business process software for manufacturers; *Companies spend up to $7 on labor and services to stitch together incompatible systems for every $1 they spend on software.*

[35] Unless one were to take the heroic position that switchover costs are constant for software from different producers.

•Network World (pg.6, 03-10-03) reported that the Gartner Research Group *found that 42 percent of CRM software licenses bought were not deployed. Despite tight expense controls, companies have been buying more CRM licenses than they can use. This inefficiency pushes total cost of ownership up 20-30 percent…*

Notwithstanding the references noted above, no one should seriously question the important and positive evolutionary effect that software has had and continues to have on business productivity and consumer utility. E-mail, word processors, spreadsheets and web browsers have fundamentally made analysis and communication more efficient and accessible. However, the development of hedonic models for software is complicated by buying motivations that may have little to do with easily defined software characteristics. Increasingly sophisticated users are more likely to be primarily influenced by factors such as whether company personnel are already trained for a particular application, whether developers are easier to find for one application relative to another, whether an application works well with existing systems or other products in the marketplace and whether significant changes to existing business processes are required.

2. Lines of Code

MGI rejected lines of code as a reliable or even reasonable alternative method for evaluating quality change in software. As with hedonics, data availability was also a problem. Without controls for advances in programming languages, compliers and the growing use of template-like code libraries; evaluations of code quantity over time are not relevant as direct measures of quality change.

3. Function Points

MGI's preferred method for valuing quality change is based on a metric called function points (FPs). FPs are basically a count of the different instances in which data can be manipulated in custom and own-account software.[36] The MGI measure reduces to a calculation of a price per FP, but MGI correctly concluded that FP data based on custom or own-account software could not be used as a proxy measure for prepackaged software (see the MGI paper listed in the references section for details). One can agree or not with the relevance of FP as a quality valuation tool, but because MGI could not locate a data source for prepackaged FPs (not surprising), their analysis does not overlap with the PPI's coverage of the industry.

4. Back to Basics

At this point the review has succeeding in confirming the obvious; developing a practical and accurate explicit quality valuation methodology for software services presents an especially difficult challenge. In many cases, it is not even possible to test an alternative methodology due to lack of relevant data. After careful review it appeared that the most promising remedy for the

[36] FPs can be thought of as various programmed operations that take in data, process it and output new data. One of the challenges presented by FPs is that it is not straightforward to comprehensively and accurately identify all of the potential operations embedded in millions of lines of code. At a minimum, a great deal of technical expertise is required and even then, the end-product is likely to be significantly judgmental.

problem of inadequate data was to make a better case to the industry as to the benefits of improved cooperation.[37] PPI staff recently visited several software producers as part of a sample augmentation project designed to adapt to rapid changes in industry outputs. During the on-site visits, producers were presented with a brief summary of how price indexes can affect measures of output, productivity and industry contributions to GDP growth.[38] Most expressed surprise when shown how measures of quality change could conceptually have a significant impact on estimates of the software industry's importance (relative to the IT sector or GDP).[39] As a result, several producers indicated a willingness to provide additional data as their products change over time. To avoid complete subjectivity and widely varied estimates for similar types of quality change, offers of seat of the pants numbers were politely declined. For instance, some producers are willing to guesstimate that a new software version is, in their opinion, 10-15 percent better than a previous version. However, no metric was offered or described as to how such an estimate could be consistently derived over time. So we went back to basics and asked if the marginal costs of new input requirements (development resources) could be provided that accounted for the changes/improvements in a current software version relative to its predecessor. As already discussed, it is the upfront development costs that are key because the marginal costs for reproducing media are generally not significant relative to transaction prices and certainly do not represent the primary inputs transformed by the production function into software outputs. Reactions to this request were mixed, but several producers agreed to estimate incremental development costs once they understood the potential for improved accuracy in official productivity numbers.

In theory, producer provided cost data for new input requirements is more consistent with the PPI's FIOPI framework and represents a conceptual improvement over direct link, direct compare or imputation. However, in practice, applying aggregate producer sunk cost data as a quality change valuation for the unit prices collected in the PPI sample is not straightforward. Recall that the simple price relative estimate used when quality improves is:

$$\frac{\mathbf{P_c} - \mathbf{VQA}}{\mathbf{P_r}}$$

Where $\mathbf{P_c}$ is the comparison period price of the new product, $\mathbf{P_r}$ is reference period price of the obsolete or discontinued product and **VQA** is the value of quality adjustment. A basic example of how explicit quality adjustment works in the PPI follows. Suppose that a screw manufacturer is repricing 1 inch steel screws sold in 1,000 lot quantities priced at 10 cents per screw in period $\mathbf{P_r}$. Market demand shifts to more corrosion resistant screws and the producer responds with new inputs, replacing 1" steel screws with 1" steel-galvanized screws priced at 12 cents in period $\mathbf{P_c}$. The PPI analyst contacts the producer and discovers that the additional cost of materials (galvanized steel) required to produce more corrosion resistant 1" screws is 1 cent. A nominal price relative (unadjusted for quality change) yields a 20 percent price

[37] Since the introduction of price indexes for prepackaged software in 1998, the PPI has had almost no success in obtaining any type of quality valuation data from survey respondents. Part of the problem is that requests to value the intellectual property that is represented in new or rearranged algorithms inevitably result in responses equivalent to "You've got to be kidding" or "We don't have the time or resources".

[38] Producers of business software almost always tout improved productivity when marketing and selling their service(s).

[39] This point was especially relevant for producers that typically do not change prices from one version to the next.

increase $\left(\frac{P_c}{P_r} = \frac{.12}{.10}\right)$, but a quality adjusted relative with a 1 cent VQA yields a real price increase of only 10 percent $\left(\frac{P_c - VQA}{P_r} = \frac{.12 - .01}{.10}\right)$. Software introduces an additional challenge for estimating VQA in that the primary inputs used in the production function are consumed up-front as development costs. To illustrate, assume a software producer spends $50 million to develop a new version of an application that (unlike screws) can be endlessly reproduced at almost no cost. In aggregate, the VQA is the $50 million sunk development costs, but the PPI survey respondents provide unit prices. If only one unit were sold, then presumably, the producer price would be at least $50 million and if 2 units were sold then at least $25 million. The producer establishes the actual unit-selling price based partly on projections of total units expected to ship. When constructing a price relative for a new software version at the unit level, the problem is how to estimate a unit value VQA. Respondents that have agreed to provide estimated development costs as a quality adjustment value will also have to provide data that enables the development cost to be apportioned across unit prices. There is no precise easily applied mechanism for this allocation, particularly in the real time production environments of a pricing agency. Continuing with the example, the producer reports that the $50 million of new input requirements have enabled a new software version that is priced at $250 per unit, which replaces a previous version that also sold for $250. A unit priced based quality adjusted relative, $\frac{\mathbf{P_c - VQA}}{\mathbf{P_r}}$, requires that the total $50 million VQA is transformed to a scale comparable to P_c. At the point of product introduction, there is no actual shipments data for the new version, but the producer can tell us the previous version shipped 5 million units and may be able to project that, if successful, the replacement version might reach 6 million units. In the first case VQA could be estimated as $50 million/5 million units=$10.00 and in the second case $50 million/6 million units=$8.33. In both cases the total sunk cost of new input requirements is transformed to a unit VQA by amortizing across actual units shipped for the predecessor or expected units to be shipped for the new version. Transforming sunk development costs for new inputs to unit values makes possible the estimation of a quality adjusted price relative expressed as $\frac{\mathbf{250 - 10}}{\mathbf{250}}$ or $\frac{\mathbf{250 - 8.33}}{\mathbf{250}}$.

Clearly, the nature of the software production function forces tradeoffs in estimating an explicit VQA based on new input requirements relative to goods-producing industries. On the other hand, recent PPI interactions with the software industry have led some producers to agree to unprecedented access to data that was previously off-limits. Part of this improved cooperation is due to the growing industry focus on productivity in their marketing materials and a new understanding of how price indexes are used as deflators. Software producers that are willing to provide cost data for new input requirements offer pricing agencies an alternative quality change valuation method that, relative to current methods, is a conceptual improvement for indexes that take an industry output perspective.

C. New Item Bias

Researchers often cite the need for more frequent (annual or quarterly) industry samples to ensure that price series continue to adequately represent current output. As outputs of the prepackaged software industry change, pricing agencies may need to adjust the content (product mix) of existing samples to reflect new services and producers. Otherwise, matched model price changes reported to a pricing agency may no longer accurately represent current market conditions. Resource and data constraints make it difficult to operationalize a strategy that could significantly reduce the PPI's current 5-7 year sample intervals.[40] A more realistic response is a targeted approach that directs scarce resources to those industries that exhibit the most rapid changes. To partly meet this challenge, the PPI developed an annual directed substitution process that enables new outputs from selected industries to be sampled from current survey respondents.[41] The actual selection of a new item is determined by a probability technique that compares a randomly generated number to the new item's market share. If the current market share of a new out-of-sample product or service exceeds the random number, then the new product or service is selected as a replacement for its predecessor. Then, depending on data provided by the producer, either explicit or implicit valuations of quality change are used to estimate a constant quality price relative between the old product and its replacement. Directed substitution has been used by the PPI for prepackaged software since 1999.

One of the limitations of directed substitution is that only changes in the outputs of current survey respondents are given a chance of selection. However, producers that did not exist or were not selected in the last sample may also have introduced new and significant software services. Therefore a more comprehensive strategy is required because price trends for the outputs of new entrants may differ from price trends reported by incumbents. The most prominent new software services entered the market as part of the post 1997 Internet expansion and range from prepackaged web-site design to web-enabled supply-chain management to e-commerce applications. Many of these new services were brought to market by companies outside the original PPI sampling frame, so without an aggressive intervention strategy they would not have a chance of selection until the next full industry resample. To reduce this delay, the PPI developed a sample augmentation process that allows new firms to be added to an existing sample.[42]

D. Customization

Some of the most expensive software licenses (often in the range of several hundred thousand dollars) are for highly specialized applications such as Enterprise Resource Management (ERM) and Supply Chain Management (SCM). The core programming code that is developed for ERM and SCM applications (and optional modules) fit the criterion for prepackaged software. But the

[40] Full economic surveys that provide detailed weighting data at the disaggregate product level are updated by the Census Bureau at 5 year intervals.

[41] An explicit directed substitution policy is of less interest if new services immediately displace or obsolete within sample services because the new services have a chance of selection in the normal item replacement strategies used by the PPI.

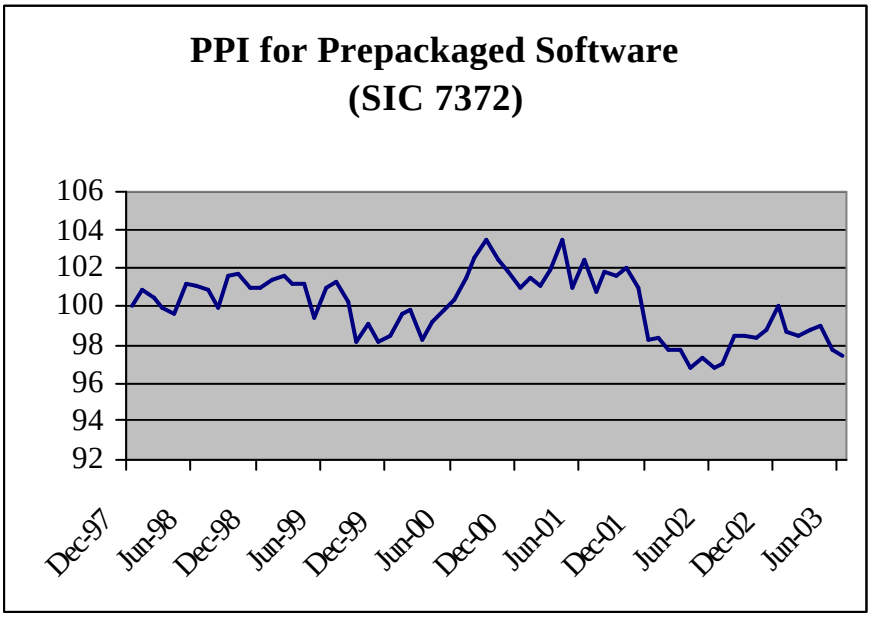
[42] Sample augmentation is a somewhat judgmental procedure that requires detailed knowledge of the firms and products/services that make-up an industry's output. There are also weighting issues that must be resolved prior to the introduction of new firms to a sample that often involve review of company specific data including financial (10K) reports, if available.

potential cost savings and organizational efficiencies promised by ERM and SCM vendors is often based on sharing and manipulating data from other proprietary custom applications that may include payroll, inventory management, production scheduling, purchasing and order entry. In other words, prepackaged applications that require links to other business process software may require back-end custom programming to establish these links. Depending on contract terms, the delivery of custom programming services can be the responsibility of the client, a 3rd party or a separate consulting arm of the ERM/SCM vendor. In all cases, custom-programming revenues is excluded from the PPI's coverage of primary prepackaged software outputs, even when required to connect in-scope prepackaged applications with out-of-scope proprietary custom applications.

IX. Time Series Data

The time series shown in the chart below presents the PPI measure of price change for the Prepackaged software industry. Because of the previously mentioned deficiency in quality valuation data, the PPI has primarily relied on direct link if new software versions are priced higher than predecessors or direct compare if new versions are priced the same as predecessors. Both direct link and direct compare result in no price change when new versions replace old versions. Therefore, the index movement shown is primarily due to measures of price change for matched models. One exception to the matched model's influence is the occasional instance where a new improved version is priced lower than a previous version. Prices are directly compared when new lower priced versions replace previous versions because while still upward biased, the amount of bias is reduced relative to direct link. The future impact of explicit measures of quality change based on the resource cost approach is uncertain as survey respondents are just starting to indicate a willingness to provide the necessary software development cost data. Part of the early and limited success in convincing respondents to increase their level of cooperation is based on directly linking improvements in productivity measures to improvements in measures of quality adjusted price indexes.

Chart 1



X. Future Trends Expected to affect the Industry

Anticipating (in a public forum) revolutionary future trends for a dynamic industry strikes me as a spectator sport best left to supremely confident and thick-skinned pundits. As I am neither, the following description of future trends are not revolutionary and by intent quite modest in scope. The primary interest in this section is to outline several recent software developments that may gain traction in the marketplace and if unaccounted for could adversely impact the accuracy of industry output price indexes.

One of the most talked about trends is the growing popularity of open-source software. Major computer producers such as IBM, Dell and Hewlett-Packard now offer servers and desktops pre-loaded with Linux.[43] If Linux continues to gain support from large computer OEMs then open-source gains additional credibility in the conservative corporate marketplace. To the extent that Linux successfully challenges Microsoft and/or Unix-based OSs some of the expected effects are greater market penetration of open-source applications such as office suites and databases. Pricing agencies with long sample intervals (several years) will need to pay close attention to the open-source movement because capturing price trends for non-proprietary software applications will likely require a sample augmentation strategy.

Another trend that may present significant price measurement challenges is the growing tendency to build software applications by linking semi-independent code modules. Applications build from modules give producers greater flexibility to tailor their products to specific customer needs. For instance, Microsoft's popular office suite (standard version) is an aggregate bundle of four applications (Word, Excel, Outlook and Powerpoint). MS Office is also offered in a professional version that adds a fifth application (Access). However, many customers may prefer a subset or an expanded bundle but have to settle for a greater or lesser feature set than desired. In 2003 Microsoft decided to greatly increase customer choice and respond to lower cost competitors such as Corel and Star office. Because the Office suite is an aggregation of modules, Microsoft was able to quickly change the composition of Office from two versions to five versions.[44] Table 4 summarizes the different application bundles available in the current Office 2003 family. Without a software architecture based on modules, it is doubtful that such a dramatic expansion of options could have been brought to market so quickly.

Table 4.[45]

[43] IBM stated that it earned more than $1.5 billion in Linux revenue in 2002 and has about 5,000 employees assigned to Linux related work. At the retail level, Walmart recently began to sell desktop computers with the Linux OS and the research group, IDC, predicts that Linux will surpass the MAC OS-X to become the number two desktop OS in the world by 2005.

[44] I am not counting a sixth version called Student and Teacher edition because it is just a lower priced standard edition.

[45] The list of Office 2003 suites and their composition was taken from the May 2003 edition of "Directions on Microsoft".

Office 2003	Word	Excel	Outlook	Powerpoint	Access	Publisher	Contact Manager	Info Path
Professional Enterprise	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Professional	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Small Business	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Standard	Yes	Yes	Yes	Yes	No	No	No	No
Basic	Yes	Yes	Yes	No	No	No	No	No

From the perspective of a pricing agency rapid expansions in bundling options made possible by the addition or deletion of modules may require sensitive reporter burden negotiations to maintain a representative product mix. On the other hand, if producers indicate that price trends for the different versions will be the same, then the need for additional reporting requirements may not be as important. In this example, price trends are likely to diverge among the different versions because of different methods of distribution. For instance, the Basic version of Office will only be sold through OEM channels and the Professional Enterprise version will only be sold through volume licensing. The remaining versions will be sold through a mix of channels including retail, OEM, Academic and volume licensing. In a related measurement issue, if applications become increasingly modular, then reporters may find it somewhat easier to estimate incremental development costs on a per module basis that could be used to value quality change.

A third trend that requires the attention of pricing agencies is the proliferation of new applications that are designed to solve very specific problems. For example, three of the most rapidly growing niche applications are for virus checkers, spam filters and intrusion detection systems. All three have grown rapidly almost entirely due to the ease of computer to computer communication enabled by the public Internet. Because the Internet is an easily accessible dynamic system that can be used in unpredictable ways, the growth of new types of narrowly focused products is almost a certainty. Pricing agencies may find that the only way to keep up with rapid introductions of new types of applications is to implement an on-going (perhaps annual) sample augmentation strategy.

There are literally hundreds of potential trends in the software market that could develop and grow quickly in importance, most of which are beyond the ability of pricing agencies to anticipate. Therefore, the best that can be expected is a reactionary response that may require adaptability due to reporter burden issues and resources sufficient to adjust, in a timely manner, sampled outputs that reflect changes in industry outputs.

XI. Need for future work

Software producers, particularly those with large market shares, have increasingly changed marketing strategies in ways that add to an already complex price measurement challenge. Many examples could be cited, but to keep the discussion manageable only two of the more interesting cases are presented.

Scenario One: Because upgrade fees are the most important revenue source for many producers, pricing agencies will almost certainly sample this type of transaction. However, if a sampled reference period upgrade fee is replaced in the comparison period with a annual subscription upgrade fee, then the pricing agency is confronted with the problem of how to construct an accurate price relative. For instance, if the original sampled upgrade fee was $400 and replaced with an annual subscription fee of $150, what is the basis for price comparison? For customers that were on a 3-year upgrade cycle, the new license terms represent a $50 increase (3 annual payments of $150=$450). For customers on a two-year upgrade cycle, the new terms represent a $100 decline (2 annual payments of $150=$300). Gartner Group recently estimated the potential costs to users when a major producer changed their upgrade license terms from point purchases to annual fees. The Gartner research concluded that *medium-sized businesses upgrading software every three years would pay anywhere from 33 percent to 77 percent more under the new plan than they did with the old. Four-year upgraders would pay 68 percent to 107 percent more*.[46] For customers that upgrade every two years, software costs would decline 19 percent. Whether the Gartner estimates are accurate or not, they help illustrate the problem of how to construct an appropriate price relative for this situation. The only unambiguous data in this example is the $400 reference period price. The unresolved question is what value should be used for the numerator of the price relative $\frac{\mathbf{???}}{\mathbf{400}}$? It is also of interest that the reference period value of $400 represents a transaction price for current output (a new software version is actually produced and paid for), but a shift to a subscription transaction in the comparison period represents a prepayment for future outputs.[47] Returning to the question of what is the proper measure of price change, it seems clear that a direct comparison ($\frac{\mathbf{150}}{\mathbf{400}}$) and the resulting 63 percent nominal price decline is inappropriate. Perhaps survey respondents that shift to a subscription based upgrade model should be asked to provide a summary estimate of the average upgrade cycle for its customer base in the reference period. If the reported average reference period upgrade cycle was 3.0 years, then buyers that previously paid $400 to upgrade will now on average pay $450 or opt out and pay full price to upgrade at a time of their choosing (or shift to a competitor's product that is still offered with a non-subscription upgrade option).[48] Unfortunately, whatever nominal price relative is constructed from available data cannot be explicitly adjusted for quality change because the corresponding comparison output has not yet been produced. A pricing agency could also consider the change in upgrade license terms as completely non-comparable and simply link-out any difference between the current period annual prepayment and reference period upgrade price. There may be other price comparison methods that are more appropriate for this example, but as is always the case, implementation will be dependent on the degree of data transparency. The correct or most realistic price relative that accounts for changes in complex licensing terms will likely vary depending on the nature of

[46] Gartner's reference to mid-size businesses is important because large business (over 5,000 employees) often enter into what are commonly referred to as Enterprise Licensing agreements (ELAs). ELAs have had at least some subscription-like features for years.

[47] A major software producer with revenues of many billions recently reported in their 10K financial report that about 26 percent of total revenues in 2002 was unearned income, most of which was due to prepayments for future software versions

[48] Full price can be anywhere from 25 to 70 percent higher than the previous upgrade price option. However, buyers may feel compelled to accept the new terms (subscription or full-price upgrades) because it is often technically difficult and costly for a business to switch to competing applications or operating systems.

the change and resulting index deflation properties. Additional research, hopefully with industry participation, is needed to develop a flexible measurement framework that addresses transitions to new pricing models and changes in right-to-use terms.

Scenario Two. Software licenses sold to the business market allow for a single CD to be installed on a server that in turn can be used to remotely install or upgrade software on multiple personal computers connected by a network.[49] If a business is expanding, then license terms usually allow IT departments to continue distributing the licensed software to new users and allow a short period for the customer to "make good" or pay for each additional installation. It is common for IT departments to establish policies that require all users to be on the same version of a particular software application. By maintaining a homogenous computing environment, cost, complexity and number of support staff is reduced. Assume that company A has settled on a particular version of an OS and decides to purchase a substantial number of new computers. Computer producers sell their desktop and notebook computers with the latest version of an OS preinstalled. If company A has standardized on an earlier OS version (likely considering current upgrade cycles of 3-4 years), then any purchase of new computers will introduce a non-standard, unapproved OS version. Or, if company A's license term for the old version OS has expired, then renewing their license with the software publisher is problematic if the old version is no longer offered. As with the computer purchase example, a new license may put company A's preferred standardized software environment at risk. Customer resistance to changes in their preferred software environment at a time not of their choosing has caused software producers to make their license terms more flexible. The new flexibility is expressed through what are often called "downgrade rights". With downgrade rights, IT departments that purchase new computers with unwanted new software versions are allowed to delete this software and install their preferred old version. Similarly, if IT departments enter into new license terms directly with software publishers, downgrade rights entitle the buyer to purchase the new version but continue to install the old preferred version. In both cases, new software versions are produced and sold, but old versions (from the originally purchased CD) are consumed. From an industry output perspective, it seems clear that when software publishers discontinue old versions, a pricing agency should price and quality adjust the new version that is actually produced and sold without regard to what actually enters the users production function. From an input perspective, the opposite holds, because input indexes are designed to measure prices for what is consumed in the users production function. Downgrade rights imply that at a minimum a temporal disconnect may exist between what is produced and consumed. Therefore, it may be useful to consider the potential effects of downgrade rights in terms of double deflation and related measures of value added at the industry level. Additional research that helps quantify constant quality differences between software produced and software actually consumed may offer useful refinements to our understanding of the current period net contribution of software outputs to downstream inputs.

XII. Conclusion

[49] Identical software sold in the retail channel often has additional copy protection code that does not allow or limits the ability to install a single CD on more than one computer. This copy protection code is removed from software sold to businesses as a concession to the need for many IT departments to distribute a software application to hundreds or thousands of users.

One of the issues raised in this review, is that in addition to the standard quality valuation problem, measures of nominal price change has its own set of challenges. Transitions from perpetual to subscription based license fees, prepayments for future upgrades (maintenance contracts) that are bundled or standalone, client access license requirements, multiple and at times exclusive distribution channels are only a few of the price measurement issues that confront statistical agencies. These issues also suggest that the availability of aggregate secondary source data, particularly scanner data from retail outlets, should be viewed with extreme caution if used to proxy a PPI. An industry output price index that is most relevant to the real producer market will proportionally represent the dominant transactions, which are to OEMs and volume license buyers. Most of the OEM and volume license transactions that characterize the software industry are opaque to secondary source data. Statistical agencies must also weigh the potential for out-of-sample bias against the resources available to quickly respond to significant changes in industry output.

The U.S. PPI coverage of the prepackaged software industry is relatively new but long-term relations with producers are being developed that should improve the transparency of complex marketing and transaction data. Resources have been committed for annual sample augmentation to adjust for rapid changes in product mix and a much improved knowledge of industry practices, based on direct interviews with large and small producers, should aid this effort.

However, even as progress is made in adapting price measures to complex industry practices, the issues of developing practical explicit quality change valuation methodologies remains. Progress on quality valuation alternatives is likely to be slow, but initial efforts to convince producers of the need for additional data, such as the amortized cost of new inputs (development costs) have been positive. Expanding producer participation in quality valuation estimates appears to require engaging survey respondents in frank discussions on the level of necessary cooperation and how price indexes can influence measures of productivity. Because software is an expression of intellectual property enabled by seemingly abstract strings of algorithms, many producers express their value proposition to customers as improved productivity. Therefore, connecting the need for quality adjusted price indexes with improved productivity measures offers one of the more persuasive arguments that statistical agencies can offer to producers. Ultimately, the growth of explicit quality change estimates will be determined by how successful the PPI makes the case for greater data transparency.

References

Carrol, J., 02-24-2003, "Hardware's Sinking Beneath a Software Sea", ZDNet, available at www.zdnet.com.com/2100-1107-985650.

Cox, J., April 28, 2003, "Users take open source databases for a spin", Network World Inc, Southborough, MA 01772-9108, Volume 20, Number 17, pg. 34, www.nwfusion.com.

DeGroot, P., July 2001, "Subscription Licensing Tested", Directions on Microsoft, available with subscription fee at www.directionsonmicrosoft.com.

Fisher, Franklin M., and Shell, Karl. 1972. "The Economic Theory of Price Indices". New York: Academic Press.

Gilbert, Alorie, 2002, "IT: New Profits in Old Glory", ZDNET News 07-12-02, available at www.zdnet.com.com/2100-1105-943494.html.

Gorko, J., Murphy, B., 1995, "A Methodological Overview of the U.S. Producer Price Indexes for Services", U.S. Bureau of Labor Statistics, Presented at the 1995 Voorberg Group Conference on Services Statistics.

Hall, B., and Khan, B., May 2003., "Adoption of New Technology", NBER working paper 9730, available at www.nber.org/papers/w9730.

International Data Corporation (IDC), 2003, "Expanding Global Economies: The Benefits of Reducing Software Piracy", Report for Business Software Alliance available at www.bsa.org/idcstudy

Kirkpatrick, David., 2003, "Ellison and Jobs: Two visions of tech", Fortune Magazine, June 26, 2003, available at www.fortune.com

McKinsey Global Institute, 2001, "U.S. Productivity Growth 1995-2000; Understanding the contribution of Information Technology relative to other factors" October 2001, Copyrighted McKinsey & Company, Inc., available at www.mckinsey.com/knowledge/mgi

Moran, P., 2003, "Developing An Open Source Option for NASA Software", NAS Technical Report NAS-03-009, NASA Ames Research Center.

Pawlak, P., 12-16-02, "Applications Require Updates for Windows.Net Server", Directions on Microsoft, available with subscription fee at www.directionsonmicrosoft.com

Prud'homme, M., Yu, K., 2002, "A Price Index for Computer Software Using Scanner Data", presented at the May 23, 2003 Brookings Workshop on Economic Measurement.

Reuters as reported in CNN-Money, 05-28-2003, "Linux Nips Microsoft in Munich".

RTI (Research Triangle Institute), 2002, "The Economic Impacts of Inadequate Infrastructure for Software Testing", Prepared by RTI for The National Institute of Standards and Technology of the U.S. Department of Commerce, www.nist.gov/director/prog-ofc/report02-3.pdf.

Triplett, J., 1983, "Concepts of Quality in Input and Output Price Measures: A Resolution of the User-Value Resource-Cost Debate", in Studies in Income and Wealth, Volume 47, University of Chicago Press for the National Bureau of Economic Research, pp. 296-311.

Triplett, J., 1986, "Economic Interpretation of Hedonic Models", Survey of Current Business, January 1986 issue, pp. 36-40, published by the U.S. Bureau of Economic Analysis.

Wilcox, J., May 2, 2003, “Analysts, No rush to new Windows”, CNET News.com, www.zdnet.com

Wilcox, J., May 16, 2001, “Windows XP: The Big Squeeze”, ZDNet, available at www.zdnet.com/filters/printerfriendly/0,6061,5083184-2,00.html.