Q/P MANAGEMENT
GROUP, INC.

Price Indexes for Custom and Own-Account Software

Final Report

Prepared for
The Bureau of Economic Analysis
by
Q/P Management Group, Inc.

June 26, 2006

10 BOW STREET
STONEHAM, MA 02180-1343

TEL: 781.438.2692
FAX: 781.438.5549
WWW.QPMG.COM

Acknowledgments

This research study was conducted for the Bureau of Economic Analysis by Q/P Management Group, Inc. Scott Goldfarb, President, Q/P Management Group, directed this research study and provided guidance on industry data and trends in software development and measurement. Roger Heller, Vice President, Q/P Management Group, assisted in all phases of the research study and provided guidance on trends in the software industry, measurement data and function point analysis. Alan G. White, Ph.D., Vice President, Analysis Group, participated in all phases of this research study and was instrumental in the research and analyses required to develop and evaluate the price indexes. Jaison R. Abel, Ph.D., Vice President, Analysis Group, participated in all phases of the research study and was instrumental in the research and analyses required to develop and evaluate the price indexes. Daniel Andersen, Senior Analyst, Analysis Group, participated in the analysis of the benchmark dataset used to calculate the price indexes.

Table of Contents

Part 1 – Discovery and Research

Project Background and Objectives

The Bureau of Economic Analysis (BEA) is the nation's economic accountant and provider of many economic measurements as part of the National Income and Product Accounts (NIPAs). The NIPAs are used for a variety of purposes in setting national economic policy and in forecasting and interpreting macroeconomic activity. They include Gross Domestic Product (GDP), which is considered the most useful summary measure of the nation's production, in addition to GDP deflators for gauging economy-wide and industry-specific inflation. They also include critical comprehensive measures of the nation's income, including Gross Domestic Income (GDI), corporate profits, and personal income.

In the 11th comprehensive benchmark revision of the NIPAs, released in October 1999, business and government expenditures for computer software were newly recognized as investment. Three types of software were identified:

- **Prepackaged software:** Software designed to meet the needs of many users with little or no customization. Examples include: PeopleSoft – Human Capital Management package, Microsoft Word and Adobe.

- **Custom software:** Software built under contract specifically for one company by a software development/consulting firm such as Electronic Data Systems (EDS), Computer Sciences Corporation (CSC) or IBM.

- **Business Own-Account software:** Software built by a company for its internal use and not for resale.

The price of own-account and custom computer software has a significant impact on the United States (U.S.) economy. Experts in the economic community have recognized that an accurate measurement of the output of the custom and own-account software is becoming increasingly important in order to measure its effect on the U.S. economy. McKinsey Global Institute documents that by 1999, non-government investment in software (or "private" software investment) grew to represent 41.6 percent of private information technology investment (in nominal terms), 11 percent of all private investment, and 1.9 percent of gross domestic product (GDP). As stated in the BEA solicitation for this project, in 2002, the nation's fixed investment in software was approximately $194 billion, or about 10 percent of total fixed investment. Of the $194 billion in fixed investment in software, about $65 billion or 33 percent was custom software. At present, there is only one quality-adjusted price index available that directly measures the prices of custom software of which we are aware.

The current price index used by the BEA to deflate custom software is estimated using a weighted average of the changes in the Bureau of Labor Statistics' (BLS) producer price index for prepackaged applications software and an input-cost index. One of the leading criticisms of the input-cost index approach is that it does not allow for changes in labor productivity.

The approach currently used by the BEA for custom and own-account software is based on combining/adjusting a number of pre-existing price indexes. The BEA has taken this approach due to a lack of a usable data set of information related to the price of custom and own-account software. The BEA has a desire to develop an index that accurately reflects the impact of productivity and quality improvements on the price of custom and own-account software. To accomplish this, a data set that includes price, productivity and data on quality attributes must be available for analysis and price index development and implementation.

The objective of this project is to develop an annual quality-adjusted price index for custom software for the period 1993 through 2003. The price index will be used to deflate both public and private fixed investment in custom and own-account software in the NIPAs.

The indexes developed are intended to be used to reflect software capability, utility, quality, productivity, price, and content over time and it is a goal to obtain a price index that controls for quality change. The price indexes should be based on an available and substantial data set that contains actual price/cost data as well as data that can be utilized to adjust for quality change.

This report represents the final of four reports that represent the total deliverables of this effort. Each report has been built on the previous version and contains the following sections:

- Part 1 – Discovery and Research
- Part 2 – Data Availability for Estimating Custom Software Price Indexes
- Part 3 – Development and Testing of Preliminary Price Index
- Part 4 – Proposed Price Index

The details within each section have been expanded upon as the project moved through to completion. The focus for this final version of the report is to enhance the materials delivered in the First, Second, and Third Interim Reports, issued on February 18, 2005, October 20, 2005, and March 8, 2006 respectively. Specifically, updated price indexes are presented in Part 3 – Development and Testing of Preliminary Price Index.

In order to accomplish the project objective, it is useful to first understand the issues and history related to the development of price indexes for software.

History of the Development of Price Indexes for Software

Given the importance of software in today's economy, a surprisingly small amount of research has been conducted that addresses the issue of software price measurement and how software prices have changed over time. The research that has been conducted has focused almost exclusively on the prepackaged segment of the software industry, even though custom and own-account software represent important segments of the overall software industry. In the following section "Academic Research" we review the academic literature on software price measurement of which we are aware, and highlight the key findings of each study. Appendix A summarizes the main findings of the existing studies of software price measurement. This research shows that both prepackaged and custom software prices have generally been declining over time, although rates of price decline in some cases differ considerably across products and time periods. We are unaware of any such studies for own-account software. In the "Statistical Agencies Practices and Procedures" section of this document, we discuss current practice software price index methodologies used by statistical agencies, but focus most of the discussion on the U.S. experience by describing software price indexes used by the BLS and the BEA. This focus is warranted given that many statistical agencies have adopted the procedures put in place by the BLS and BEA. In the "Academic Research" section of this paper we draw out the key implications of the academic and government research on software price indexes for initiatives for developing and improving upon price index measures for custom and own-account software in the U.S.

Academic Research

Prepackaged Software

While relatively few studies measuring the prices of prepackaged software exist, much has been learned on this topic during the past decade of research. An early effort by Oliner and Sichel [1994] employs matched-model price index methods to control for quality change over time by intertemporally comparing prices for only similar software products. Using price quotes collected from various personal computer magazines for the period 1985 to 1993, they find Annual Average Growth Rates (AAGR) declines in prepackaged software prices of 2.6, 4.5, and 4.7 percent for word processors, spreadsheets, and databases, respectively. While this initial contribution raised the issue of quality change in software price measurement, it stopped short of actually doing so in a complete manner since the use of matched-model methods do not explicitly adjust prices for changes in quality.

Estimates of quality-adjusted price declines for the U.S. prepackaged software market using hedonic regression techniques were first reported in Gandal [1994, 1995], Brynjolfsson and Kemerer [1996], and McCahill [1997].[1] Over the 1986 to 1991 time period, Gandal [1994] finds declines in hedonic quality-adjusted prices for spreadsheets of 15 percent per annum,[2] generally consistent with the final declines reported by Brynjolfsson and Kemerer that lie in the range of 14.8 to 16.5 percent per annum covering 1987 to 1992, and by McCahill in the range of 9.0 to 16.9 percent per annum for the period 1986 to 1993.[3] McCahill also reports declines in quality-adjusted prices for word processors in the range of 15.1 to 18.5 percent annually, while Gandal [1995] reports a smaller decline of 1.5 percent annually for databases over the period 1989 to 1991.

During this time period, similar studies were conducted by German researchers utilizing non-U.S. prepackaged software sales information. Grohn [n.d.] reports hedonic estimates of declines in quality-adjusted price indexes for word processors in the range of 11.3 to 36.9 percent per annum for Germany over the 1985 to 1995 time period. Harhoff and Moch [1997] also analyze prepackaged software price trends in Germany from 1986 to

1 Much of the research involving the estimation of hedonic price equations for prepackaged software has focused on identifying and measuring the presence of network effects. However, it is unlikely that this focus will be central to the construction of a price index for custom and/or own-account software. Therefore, this section focuses on the price trend results obtained from the price index resulting from such analysis.

2 Gandal [1995] also finds declines in quality-adjusted prices for spreadsheets of 4.4 percent per annum over the shorter time period 1989 to 1991.

3 Earlier estimates by Brynjolfsson and Kemerer, cited by Oliner and Sichel, reported considerably smaller rates of price decline.

1994. Based on price quotes from German personal computer magazines, they compare price trend results using both matched-model and hedonic price index methods. Interestingly, they report a decline in their price index for databases of 9.25 percent per annum using matched-model methods, a greater rate of price decline than that found based on hedonic methods (7.41 percent). Harhoff and Moch attribute this finding to the fact that older versions of database software were sold at greatly reduced retail prices as newer versions were introduced to the market.

A limitation of much of the early software price measurement research is the reliance on retail list price information. Prud'homme and Yu [2002] address the issue of list price by utilizing Nielsen scanner data for software sales in Canada. They report price declines in transactions prices for an aggregate of all prepackaged software in the range of 4.4 to 7.9 percent per annum using matched-model price indexes for the period 1996 to 2000, and for a substantial number of sub-aggregates covering a wide range of prepackaged software products, including games. Two prominent findings in Prud'homme and Yu are that rates of price decline differ considerably across various types of prepackaged software (high for games, smaller for operating systems), and that because of rapidly changing market shares, growth rates of price indexes differ substantially (even, in some cases, in sign) depending on which weights are used.

Abel, Berndt, and White [2003] compute matched-model price indexes for prepackaged software products sold in the U.S. by Microsoft at the first line of distribution, which allows for the inclusion of sales to both the finished goods (i.e., distributors and resellers) and original equipment manufacturers (OEM) channels of distribution. Thus, theirs is the first study to incorporate information from distribution channels other than retail outlets, which for a company like Microsoft represents the vast majority of prepackaged software sales. Their price index calculations also incorporate numerous product forms that exist in the prepackaged software industry, including full versions, upgrades, software suites, and volume-licensing sales. Using a matched-model price index, Abel, Berndt, and White report average annual declines in prices from 1993 to 2001 for Office (4.78

percent), Word (10.64 percent), Excel (8.17 percent), and personal computer operating systems (0.39 percent).

The early literature on hedonic price indexes for pre-packaged software began with stand-alone applications. Over time, however, such applications have increasingly been sold as components of productivity suites. In addition, the software products with which these applications interface – personal computer operating systems – were ignored in the early hedonic price index literature.[4] White, Abel, Berndt, and Monroe [2004] extend the early literature on hedonic price measurement to include personal computer operating systems and productivity suites. Based on price quotes from *PC World* magazine, they report quality-adjusted price declines ranging from 15.72 to 17.52 percent for personal computer operating systems over the 1987 to 2000 time period and ranging from 14.97 percent to 16.25 percent for personal computer productivity suites over the 1984 to 2000 time period, although the authors note that rates of price decline are larger in the last half of the sample than in the first half for both product categories. White, Abel, Berndt, and Monroe [2004] also compare their hedonic price trend results to those derived using a matched-model approach, and report finding significantly smaller price declines when quality change is not explicitly addressed.

Custom Software

To the best of our knowledge, the only price indexes that have been developed for custom software, other than that published by the BEA, are provided in a recent unpublished study by Ethiraj, Kale, Krishnan, and Singh [2004]. According to the authors, the major advance of this research is the explicit movement away from the cost-plus focus that exists in the literature on the economics of custom software development to understanding custom software pricing from a demand-side perspective. Doing so allows one to focus on the value of the software from the consumer's perspective, which permits the use of hedonic techniques. Indeed, the authors argue that the pricing of custom software projects is broadly determined by three factors: *project specifications* which

[4] Using an alternative quality-adjustment methodology, based on counts of function points, McKinsey Global Institute [2001] reports a price decline of 11.7 percent for Microsoft's operating systems from 1988 to 1998.

determine the input costs, *contractual terms* which determine the risk allocation between vendor and client, and terms of the service level agreement which determines expected product and service dimensions.

Ethiraj, Kale, Krishnan, and Singh develop a hedonic pricing model in which price, defined as the revenue in USD received from each project in their data, is determined by three broad factors: project characteristics, project governance, and project quality attributes. The specific variables used to represent project characteristics include: project size (measured by function points), project team size, project duration, project team experience, vendor experience with client, project team education dispersion, client industry domains, development platforms, and time. The specific variable used to measure project governance is contract type, which differentiates between time and materials contracts and fixed-price contracts. The specific variables used to represent project quality include: in-process defect density, effort overrun, and schedule slippage.

Ethiraj, Kale, Krishnan, and Singh report an average annual price decline of 14.5 percent for the custom software projects included in their sample during the 1999 to 2002 time period. Interestingly, over this same time period the authors report average annual quality-adjusted price decline of 5 percent for custom software projects under a time and materials contract and an average annual quality-adjusted price increase of 9 percent for custom software projects under a fixed-price contract. A noteworthy limitation of this initial custom software study, which is recognized by the authors, is its reliance on data for 160 largely outsourced projects from a single software development firm in India.

Statistical Agency Practices and Procedures

The U.S. BLS and BEA have long recognized the importance of developing accurate measures of price change for software, and have led the way in undertaking research initiatives to improve the measurement of changes in the prices of software over time. Indeed, other statistical agencies have followed the lead set by the U.S., and below we describe the software price indexes of Canada and Australia as being representative of procedures adopted by various statistical agencies around the world on deflators for software. A more complete summary of these procedures can be found in Table 3.5 of the Report of the OECD Task Force on Software Measurement in the National Accounts. The discussion below focuses primarily on the current practices of the BLS and the BEA.

United States

The BLS first began publishing a producer price index for prepackaged software in December 1997, based on a survey of prices charged by software manufacturers at the first line of distribution. The prices collected are intended to represent sales to OEMs (such as Compaq) as well as distributors (such as Ingram). The current methodology of the index is a monthly weighted matched-model price index, where prices of similar products are compared over time. The company weights (reflecting the relative importance of sales of the manufacturers represented in the index) are updated every five to seven years. Over the period December 1997 through December 2004, the BLS producer price index for prepackaged software declined at an annual rate of approximately 1.02 percent.[5]. Over the same time period, the BLS consumer price index showed that prices paid by consumers declined at an annual rate of 6.80 percent, based on a similar methodology to that used in the producer price index. The BLS also publishes a number of sub-indexes of its prepackaged software producer price index, including applications software (sold in a suite and stand-alone), and games and other prepackaged software.

The BEA also computes software price indexes to serve as deflators, i.e., for the purposes of converting current dollar measures of gross domestic product into real (inflation-adjusted) measures, and for measuring the output of the U.S. economy in the aggregate, and by industry sector. The BEA currently computes three separate price indexes for software, one for each of prepackaged, custom, and own-account, by using a variety of indexes published by the BLS and by drawing on the body of academic, and other research on price indexes for prepackaged software.

[5] This producer price index has been renamed to "software publishers."

The prepackaged software price index is taken directly from the BLS price index for prepackaged software. As Grimm and Parker [2000] note, the literature on hedonic quality-adjusted price indexes for prepackaged software documents that quality-adjusted prices for software applications typically have fallen more rapidly than corresponding matched-model price indexes. This arises since matched-model indexes do not fully capture many of the quality improvements that have occurred in software applications over time. A discussion of some of these quality improvements in the context of personal computer operating systems and productivity suites is found in White, Abel, Berndt and Monroe [2004].

In recognition of the limitation of matched-model indexes in adjusting prices to reflect quality changes over time, the BEA currently makes a "bias adjustment" to the BLS producer price index for prepackaged software, based on available research that allows a comparison of price change between the matched-model and hedonic techniques. The adjustment is based on a comparison of the matched-model price indexes of Oliner and Sichel [1994], and a BEA hedonic price index for spreadsheets and word processors, over the period common to both analyses, 1985 to 1993.[6] The average annual difference between these two sets of price indexes over this period is -6.3 percent. The BEA applies one half of this annual adjustment (or -3.15 percent) to its deflator for prepackaged software. In other words, the BEA deflator for prepackaged software declines 3.15 percentage points greater than the BLS prepackaged producer price index in any given year. While this may reflect the best method currently available, the hedonic results of White, Abel, Berndt and Monroe [2004] show declines in the prices of operating systems and productivity suites of 14.96 and 10.42, respectively, between 1985 and 1993. These price changes represent approximately 7 and 10 percentage points greater decline than corresponding matched-model indexes, respectively, suggesting that the 3.15 percent adjustment used by the BEA may have been a conservative estimate of a quality adjustment.

[6] The BEA index is itself an extension of work by Gandal [1994], Brynjolfsson and Kemerer [1996], and McCahill [1997], all discussed in Academic Research/Prepackaged Software section of this report.

The BEA also constructs two other price indexes for software, each based on a composite of other indexes, rather than a direct index construction. The price index for own-account software is based on a weighted average of two indexes: a compensation (wage) index for computer programmers and system analysts and an index of overhead associated with their work. Weights are assigned equally to both. Separate compensation indexes are used for government and business to recognize the fact that compensation rates in the two sectors may have moved somewhat differently over time. One drawback recognized by the BEA is that the compensation index component of the BEA is based on an assumption of *constant* labor productivity, i.e., workers are assumed to produce a constant level of output over time. While this is a drawback, the BLS has adjusted their producer price index for prepackaged software to partially address this issue. The BEA price index for custom software is a weighted version of the own-account software price index and the prepackaged software producer price index. The weights are 75 percent for changes in business own-account software and 25 percent for changes in prepackaged software prices.

The BEA undertook a preliminary analysis to assess the feasibility of implementing a quality-adjusted price index for own-account software based on function points. According to Wasshausen [2003a, 2003b], this initiative was abandoned since it was found to be unsatisfactory for the following reasons: 1) there was a lack of price data, one of the key components of constructing a reliable price index; 2) the data were not sufficiently rich or large to produce robust estimates of price change; and 3) there was a feeling that function points alone did not fully capture the true quality of the underlying software. In Part 2 – Data Availability for Estimating a Quality-adjusted Custom Software Price Indexes, we discuss each of these drawbacks in the context of the data set we use to construct our proposed price indexes.

Non-U.S. Statistical Agencies

The current practice for Canada mirrors that of the U.S.[7] Currently there is no pre-packaged software price index, and an adjusted version of the U.S. BEA price index is used. This price index used to deflate prepackaged software investment is an average of the BEA index, weighted by the domestic share of supply on the domestic market, and an exchange rate-adjusted version of the BEA index, weighted by the import share of supply to the domestic market. The price index for own-account software, likewise, mirrors that of the BEA, although compensation and input cost indexes produced by Statistics Canada are used in forming the weighted average. The custom software index, as in the case of the U.S., is a weighted average of the prepackaged and own-account software price indexes, using the same set of relative weights.

The Australian Bureau of Statistics currently does not have a price index for custom or own-account software, although they are currently undertaking a project to investigate feasible methods for implementing a price index, including a consideration of function points. A list of research currently being conducted by statistical agencies is discussed by the Ottawa Group, an inter-statistical agency organization set up to promote research on various measurement issues faced by statistical agencies.[8]

Implications for Current Project

The past decade of research by both academic researchers and government agencies has greatly increased our understanding of software price measurement, particularly in the area of prepackaged software. Given the differences that exist in both production and consumption between prepackaged software and custom and/or own-account software, it is important to extend this level of understanding to the custom and own-account

[7] A discussion of these procedures can be found at http://www.statcan.ca/english/freepub/13-605-XIE/2003001/conceptual/2001software, "Capitalization of Software in the National Accounts," visited 1/31/2005.

[8] http://stds.statcan.ca/english/voorburg/aspfiles/2004-subsubjectsearch2.asp?criteria=Computer%20Services&criteria0=Prices.

segments of the software industry. Below are the primary implications for such a project that arise from the existing software price measurement research:

- The key issues that emerge from the various research initiatives undertaken by statistical agencies and academics is that quality improvement in all types of software is pervasive, and that current practices fail to adequately incorporate such quality improvements into measures of price change. Existing research strongly suggests that failure to explicitly incorporate measures of quality change understate (overstate) price declines (rises).

- The current custom and own-account price indexes, based on the assumption of constant labor productivity of programmers, are problematic. Furthermore, the index is based on a composite of other indexes, rather than a direct calculation of price change.

- The pre-existing literature on software price measurement and the recent addition to this literature of a quality-adjusted price index developed for custom software suggests that implementing similar analyses for custom and own-account software is feasible.

- The focus of network effects that is central to prepackaged software price indexes is not likely to be relevant to custom or own-account software projects.

- Data limitations have hampered research on price indexes for custom and own-account software. A sufficiently large data set spanning a number of years that contains reliable price information, in addition to quality attributes beyond estimates of the development costs associated with the projects, is the key to successfully implementing quality-adjusted price measures for custom and own-account software. Indeed, the lack of price data was a major roadblock in constructing a price index for own-account software (Wasshausen [2003a, 2003b]).

Part 2 – Data Availability for Estimating Custom Software Price Indexes

Database Availability and Overview

As described in Part 1 of this report, establishing a new price index for custom software is highly dependent on an appropriate data set. The data set should be sufficient in size, span a number of years and contain information on both price and quality attributes. One approach would be to construct the data set in a way that it would be comprised of similar products year-to-year in order to compare price changes over time from a matched-model type perspective. We know, by definition, custom and own-account software products are by their nature unique. Therefore, care must be taken when making price comparisons between them. There is, however, a potential common unit of measure for software products that can be utilized as a basis for price comparison. This unit of measure is function points. Function point analysis (FPs) is a worldwide software industry standard for measuring the output of the software development process. Function points measure software size from the user's point of view and also factor in quality changes that improve software use from the customer's perspective. A more detailed overview of FPs is provided later in this section.

Many companies around the world use function points in conjunction with other data in order to measure software productivity, quality and price. These data are often made available to organizations that are involved in establishing performance measurement databases for the purposes of benchmarking and estimating software development. The majority of these benchmark databases are the proprietary property of privately held companies (see Appendix B). One such database is maintained by Q/P Management Group, Inc. (Q/P) and is used for this project.

Q/P's benchmark database is used by companies, commercial software developers and government agencies around the world. Q/P has been involved in numerous client studies to establish quality and productivity measures, baseline development environments as well as to establish the value of software and estimate the cost, schedule and effort to deliver software projects. Q/P’s benchmarking methods are comprehensive and are based on industry-accepted practices. Q/P’s database is comprised of thousands of projects and applications from over 100 organizations. The majority of these organizations are Fortune 1000 companies many of which have used the data for their own internal use. These organizations have utilized benchmark productivity, quality and cost/price data for internal performance comparisons, vendor performance comparisons, bid evaluations, and estimation of the cost, schedule and effort to deliver future projects.

Q/P Database Components

Q/P’s benchmark database contains statistics on software projects collected from 1993 to 2004. An analysis of the available data was performed to remove projects that were considered to be unsuitable for the establishment of a price index. This analysis removed 1,267 projects and resulted in a dataset containing 5,651 new development and enhancement projects representing 12 industries. See Exhibit 1 for descriptive statistics on the dataset. The dataset went through the extensive analysis to ensure that the data being analyzed are accurate and representative of industry activities occurring during the timeframe represented. Data preparation and clean-up activities included:

- Elimination of obvious project outliers that would unduly influence trends and statistics
  - Price per function point outliers >\$6,000 and < \$50 (eliminated 450 projects from the original database.)
  - Extremely high or low productivity projects based on size and platform (eliminated 154 projects)
- Elimination of projects with critical incomplete data, i.e. projects that contained function point data but lacked effort and/or cost data. (eliminated 551 projects)
- Elimination of individual data elements related to schedule days and defects that were significant outliers in the dataset.
- Elimination of non-USA projects (eliminated 42 projects)
- Elimination of projects that were software package implementations (eliminated 70 projects)

In addition to the elimination of projects and data elements, values were assigned to missing data elements that could be reasonably estimated based on the company/platform averages, company averages or annual averages. The data elements where values were assigned when missing were the Value Adjustment Factor and Defects per Function Point. It should also be noted that Project Price/Cost in most cases was calculated by multiplying project hours by a fully loaded hourly rate (labor, overhead and profit when appropriate). This is in fact the basis for determining price/cost in almost all projects in the software industry. With the exception of only a few projects, the price for custom software projects is based on an hourly rate that the outsourcer charged the customer multiplied by the number of hours charged against the project. This hourly rate was provided to Q/P by the outsourcer or customer and is based on the rate that was negotiated between the customer and the outsourcer. The negotiated rates are generally competitive across outsourcers and therefore do not generally vary by more than 20% when comparing one outsourcer to another. The rates are unique for each client in the database and typically apply to all of the projects for a company in a given year. The rates usually change for each contract year as predetermined by negotiated terms of the outsourcing agreement. Outsourcing contracts are typically for a period of three (3) to ten (10) years. In some cases for own-account software projects the fully loaded hourly rate was derived by applying overhead to the company labor only rate or by applying the annual average fully loaded rate. The finalized updated benchmark database provides the basis for the construction of the proposed price index. **Exhibit 1** summarizes the occurrences of projects by year.

The benchmark database is comprised of a mix of outsourced and in-house development projects.[9] The outsourced projects are based on contracts where the outsourcer is delivering custom software for their clients for a contracted price. Outsourcing became more common after 2000. As a result, the benchmark data prior to 1999 contains more

[9] The benchmark database contains a limited number of projects that utilized offshore resources to perform some of the project activities. Currently these resources have a minimal impact on project prices. The use of offshore resources should be addressed in the future when more projects utilize more offshore resources increasing the impact on price.

own-account development projects than outsourced projects. For this reason, the resulting custom software price index may be based in-part on an analysis that includes own-account software projects. A separate analysis of own-account software projects may also result in a separate price index for own-account software

Exhibit 2 further describes the data elements in the benchmark database and how the elements are used as software measures. The implications of these variables for the purposes of constructing a new price index for custom and own-account software are also highlighted in Part 3 of this report.

There are a number of data elements listed in **Exhibit 2** related to FPs. These include the Unadjusted Function Points, Value Adjustment Factor and Adjusted Function Points. These data elements related to software size are very important aspects of a statistical implementation of a hedonic model. The following describes the concepts behind FPs and how FP counts are established.

Overview of Function Point Analysis

Function points have been recognized as an international standard for sizing application software. There are IFPUG affiliates in Brazil, South Korea, Italy, Australia and the UK. Function points are utilized by many Fortune 100 companies in the United States and have been recognized by ISO as an accepted software functional sizing metric. This worldwide popularity has led to the use of function points in many aspects of managing the software development process.

FPs are being used by many companies to measure the productivity of their software development processes and the quality of the products they produce. Outsourcing agreements are specifically written to include benchmark studies based on function points to evaluate the performance of the outsourcers in terms of productivity (FP/hour), price/cost ($/FP), quality (defects/FP) and project schedule. Custom software development companies have utilized function points to determine a fair market price to charge for their solution. Data sets exist in the software measurement industry that contain function point, price/cost and quality statistics. These data sets are mostly proprietary, but offer a unique opportunity to be utilized in developing more accurate price indexes for custom and own-account software.

Function points are a unit of measure that quantifies the size of a software application or project. Conceptually, FPs are similar to how square feet are used to measure the size of a house. In addition to measuring size, FPs are extremely useful in estimating the cost, schedule and effort to deliver software projects, as well as to help manage change of scope, measure productivity and communicate functional requirements.

One of the initial design criteria for FPs was to provide a mechanism that both software developers and users could utilize to define functional requirements. It was determined that the best way to gain an understanding of the users' needs was to approach their problem from the perspective of how a computer can help a user do their job. Therefore, one of the primary goals of FPs is to evaluate a system's capabilities from a user's perspective.

To achieve this goal, the analysis is based upon the various ways users interact with computerized systems. From a user's perspective a system assists them in doing their job by providing five (5) basic Function Types. Two of these address the data requirements of an end user and are referred to as Data Functions. The remaining three address the user's need to access data and are referred to as Transactional Functions.

The first data function is known as an Internal Logical File (ILF). Its function is to store the business data that the user organization is responsible for maintaining. The second data function is called an External Interface File (EIF). As its name implies, this file is outside of the application being counted. While it contains business data, the file contents are maintained by a different application. In practice, the application being FP counted can only reference this data. Each logical group of data used by the application is categorized as either an ILF or EIF.

Once the data needs have been addressed, the next step is to categorize the various transactions the application provides the users. The first transaction is referred to as an External Input (EI). The purpose of an EI is to maintain the data on the files owned by the application (ILFs). The second and third transactions address the user's needs to receive data from the application. One of these transactions is called an External Inquiry (EQ). This transaction allows the users to directly retrieve stored raw data from the files (ILFs and EIFs). The last transaction is an External Output (EO). The primary difference between an EO and an EQ is that the data contained in an EO is transformed from the raw data on the files (ILFs and EIFs) through derivation or mathematical logic contained within the application.

When counting the FPs of an application, the counter maps each of the user recognizable data files and transactions to one of the five Function Types. The mapping is determined based on a set of standard rules that are maintained by the International Function Point Users Group (IFPUG). Each of the functions is assigned a functional complexity. Functional complexity is an indicator of how complicated a function is from a data perspective and it is used to determine how many unadjusted function points (UFPs) will be assigned to the function. The number of UFPs assigned is determined by standards set by IFPUG. **Exhibit 3** represents the IFPUG standard number of UFPs that can be assigned for each function type.

The UFP values of each file and transaction are added together to determine the UFP size of a project or an application. For example, if a project developed one high complexity internal logical file (15 UFPs), one high complexity output report (7 UFPs) and two low complexity inquiries (3 UFPs each) the resulting UFP count would be: 15 + 7 + 6 = 28 UFPs.

There is an additional component of FPs known as the General System Characteristics (GSC) that is used to adjust the UFP count described above. This adjustment is based on specific characteristics inherent to the application and it accounts for how an end user interacts with the software. In this sense, it can be thought of as a quality-adjusted factor that is applied to the FP count. **Exhibit 4** depicts the 14 General Systems Characteristics that are evaluated.

The GSC's are used to calculate the Value Adjustment Factor (VAF) which is applied to the UFP count to calculate the Adjusted Function Point (AFP) count. The VAF can modify the UFP count by +/- 35%. Generally speaking, an application that has a low VAF tends to provide a low degree of usability to an end user and is of lower quality than one with a higher VAF. Legacy batch applications tend to have a low VAF. Applications with a high VAF tend to provide a high degree of usability to an end user. Web based applications tend to have a high VAF. The AFP may be thought of as a measure of "effective" Function Points, in the sense that the measure explicitly adjusts for other quality factors of a software project. Usability factors include:

- Navigational aids
- Colors
- Dropdowns
- Interfaces
- Menus
- Help
- Minimal screens
- Video interface
- Automated cursor movements
- Multiple input/output devices
- Flexible reporting
- User maintenance of business control rules/data
- User control of edits and validation of data
- Response times
- System availability/reliability
- Hardware/software compatibility
- Mouse interface
- Multi-lingual

Many of these factors are considered in the General System Characteristics evaluation and the resulting VAF. Therefore the VAF might be used as a quantifiable number that can be used as a quality adjustment factor in a hedonic regression.

Summary of Data Availability and Usage

Over 5,500 projects from 1993 to 2004 were analyzed in this study. Due to the timing of this study, the 2004 data were incomplete in terms of the number of projects collected. For this reason, 2004 data were eliminated from the price index analysis. In future years, additional data from 2004 and beyond can be made available for updating price indexes over time, if necessary. The database contains a variety of variables, including: function point counts, level of effort, cost/price data, defect information, schedule duration, industry type, computing platform, and other data elements that will be utilized as variables. This information along with the appropriate statistical analysis was used to develop the Preliminary Price Index as discussed in the following section Part 3 – Development and Testing of Preliminary Price Indexes.

Part 3 – Development and Testing of Preliminary Price Indexes

Overview of Development of Preliminary Price Indexes

Using the Q/P Management database described in Part 2 above, two general approaches are proposed and are now evaluated in developing preliminary price indexes for custom and own-account software. The first approach utilizes a traditional matched-model type methodology that is based on matched software projects to compare price changes over time for similar software projects. This approach does not adjust prices explicitly for changes in quality, but rather controls for quality changes by making price comparisons between similar groups of software projects over time. The second approach involves computing quality-adjusted price indexes based on hedonic regression models that explicitly attempt to adjust for variation in quality across projects and changes in project quality over time. We report the results of traditional hedonic price indexes as well as biennial chained hedonic price indexes.

Another potential approach that modifies the existing method utilized by the BEA to calculate custom and own-account software price indexes was evaluated in the Second Interim Report, but was not further developed here in order to focus on the two general approaches described above. This alternative is discussed in **Appendix C**.

Descriptive Statistics and Simple Average Price Trends

Before turning to the preliminary price indexes that have been computed, it is useful to provide descriptive statistics and simple price trends based on aggregate annual average prices or prices per adjusted function point. **Exhibit 5** provides selected descriptive statistics for some of the key variables in the Q/P Management database. For numeric variables, the mean, minimum, and maximum values are reported. For discrete variables, frequency tables are presented.

The composition of the Q/P Management database and how this has changed over time is highly relevant to the development of preliminary price indexes for custom and own-account software. Because these data represent the custom and own-account software projects to which Q/P Management has provided benchmarking and consulting services over time, there is considerable year-to-year fluctuation in the composition of software project types, industries, and platforms covered in the data Q/P Management has complied over the years. This is due to the changing mix of clients and industries Q/P Management Group has done business with over the years. For example, **Exhibit 6A** shows the distribution of custom and own-account software projects by Industry from 1993 to 2003. Projects from twelve separate industries are captured in the Q/P Management database – however, there are gaps in industry coverage in the data over time. For example, financial services appear to be prevalent in the 1990s, while telecommunications appears to be prevalent after 2000. **Exhibit 6B** shows the distribution of custom and own-account software projects by software type over time. Importantly, a rapid shift from own-account toward custom-outsource projects appears to have taken place between 1999 and 2001 in these data. This is due to the trend of increased use of software development outsourcing in more recent years.

The implication of this volatility for the current analysis in the context of a matched-model price index is that one cannot consistently follow price trends for similar software projects in a given industry for a particular type of software over time. As explained below, both industry and software type are among the important dimensions in defining the elementary units in a matched-model price index on which software project price comparisons are ultimately made. Such volatility may also have important implications for the construction of hedonic price indexes. In particular, the traditional hedonic regression model assumes parameter stability over time. If such an assumption does not hold in practice, more flexible regression models may be needed. To address this issue, we report the results of both traditional hedonic price indexes and hedonic price indexes computed by chaining the price changes based on coefficient estimates on time dummy variables from sequential sets of biennial regressions.

Exhibit 7A shows a simple aggregate price trend for custom and own-account software over the 1993 to 2003 time period. Without accounting for quality change, prices appear to have increased annually by over 26 percent. As a first step toward accounting for quality, **Exhibit 7B** shows a similar trend using price per adjusted function point, rather than price. Making this adjustment reduces the annual rate of change from over 26 percent to approximately 13 percent. A fuller discussion of function points is contained in Part 2 above.

Matched-model Price Indexes

In traditional matched-model price indexes, prices of similar products are compared in adjacent time periods. In this approach the elementary unit which serves as the basis for price comparisons involves a measure of function points – specifically price per adjusted function point ($/FP) calculated across a variety of different dimensions. To ensure that like comparisons are being made, the elementary unit will be further refined and disaggregated by industry, platform, size and project type (new development or enhancement). This more detailed breakdown and matching criterion is required in order to factor in the differences in cost and productivity based on the industry, the software development environments of different platforms (mainframe (MF), web, client server (CS) and mixed), the size of the project based on the number of function points and the differences between developing new software and enhancing pre-existing software. Furthermore, these four factors are typically considered when benchmarking software in order to provide a more accurate comparison between an organization's software performance and industry averages.

The key to constructing a matched-model price index is first to identify the elementary units for which prices are computed and compared over time. In the context of custom and own-account software, prices of 'similar' projects should be compared over time. In constructing the boundaries of what we consider 'similar' we have adopted a number of dimensions in grouping software projects together, including software type (e.g., own-account, custom-outsource), project type (new development or enhancement), CPM version (i.e., the underlying function point counting method), CMM level, platform (e.g., mainframe, client server), and industry (e.g., telecom, finance). We considered each of these dimensions to show sufficient variation in terms of cost, productivity, and quality, that it warranted grouping the software projects into these distinct groups. A discussion of these different dimensions is discussed in Part 2 above. By way of comparison, examples of matched-model price indexes for personal computer prepackaged software have been reported by Oliner and Sichel [1994], Prud'homme and Yu [2002] and Abel, Berndt, and White [2003].

In the context of personal computer operating systems, a matched-model price index would compare, for example the prices of a full version of Microsoft Windows 98 over time, but the prices of Microsoft Windows 98 would not be compared to the prices of Microsoft Windows XP Professional, since these products would be considered to be different. Custom software projects, by their nature, are unique, as each is designed to meet the software needs of a variety of different types of customers with varying software needs. As such, a matched-model index should be based on comparing the prices of *identical* software projects over time (i.e., for the *same* company for *similar* projects). Since this is not feasible (given that no matches would be found over time for a specific project), we have grouped projects into elementary units where the elementary units are designed to group projects with similar quality/productivity characteristics together.

We compute a chained matched-model Fisher price index with updated annual weights, where the weights are the number of software projects used in computing the average price per adjusted function point for each distinct elementary unit. Due to a lack of matches by industry over the time period 1993 to 2003, we grouped the industries into three broader categories based on each industries' software needs: Group 1 (Public Utilities, Telecom, Pharmaceutical, Software Developer, Government), Group 2 (Financial Services, Insurance, Education, Health Care), and Group 3 (Retail, Manufacturing, Transportation). This grouping was established based on industries with similar software development productivity rates due to like business constraints and business functions. For instance, Group 1 industries are heavily regulated with demands for high quality software and comprehensive software documentation. Group 2 companies are generally service organizations and Group 3 companies have similar functional areas such as inventory control and distribution. The resulting matched-model price index, which controls for project type, software type, CMM level, platform, and industry grouping rises at 22 percent per annum over the 1993 to 2003 time period, as shown in **Exhibit 8A**.[10] We also computed an alternative Fisher matched-model price index where an additional category, project size (based on adjusted function point count) was added in defining the elementary unit. As seen from **Exhibit 8B,** the resulting CAGR decreases from 22 to 19 percent per annum. These results compare with a greater increase of over 26 percent per annum for the aggregate annual price of custom and own-account software. Note, however, that this simple aggregate price suffers from the drawback that it masks considerable heterogeneity among the software projects over which the average is formed – aggregating over projects that differ possibly in terms of features, design, and overall functionality.

When one computes an aggregate price per adjusted function point (and thus making an adjustment for function point count), the rise in the price of custom and own-account software falls in half from approximately 26 percent per annum to approximately 13 percent per annum. Note that controlling for quality change and making price comparisons over comparable software projects over time with a matched-model price index, results in price increases that are above those of the average price per adjusted function point. However, great care should be taken when comparing the results of these matched-model price indexes to other measures of price change since they are constructed with only a subset of the available data due to matching issues. The availability of matches over time allows the matched-model price index to make price comparisons over comparable projects over time and thus control for project quality. Gaps in data coverage (e.g., no software projects in given industry groups or for given platforms over consecutive years) makes a matched-model price index less reliable, and also more difficult to and interpret, given that many imputations may need to be made for years in which data coverage is thin.

[10] Due to data limitations, we were not able to use the variable CPM version when constructing the elementary units for these preliminary matched-model price indexes. This variable has been included in the preliminary hedonic price analysis that follows.

As seen in **Exhibit 6A**, software projects in the Telecom industry dominate in the sample from 2001 forward, although there were no Telecom projects in the years 1993 to 1995. Analogously, in **Exhibit 6B** the growing importance of Custom-Outsource software projects is apparent over time while Own-Account projects appear to become less relevant over time. The implication of **Exhibits 6A** and **6B** is that, if elementary units are formed along dimensions across which there is continual change, the challenge of finding matches becomes increasingly difficult. In fact, of the approximately 5,500 software projects in the database, only half are used in computing the matched-model price indexes shown in **Exhibit 8A** and **Exhibit 8B**. This lack of coverage may explain the relationship between the matched-model price indexes and the simpler aggregate price per adjusted function point. In addition, because of the difficulty in forming consistent elementary units over time, other potentially relevant price determining factors, such as productivity, can not be incorporated into the matched-model price indexes. However, as shown in **Exhibit 9**, productivity, measured by adjusted function points per hour, has declined by nearly 8 percent per annum over the 1993-2003 time period. Thus, controlling for such changes appears to be an important element when constructing price indexes for custom and own-account software.

Much of the reported price increase can be attributed to the marked productivity slowdown post-2000, as organizations focused on higher quality software projects at the expense of productivity gains. Throughout the 1990's many software organizations were focusing on delivering software as fast as possible to the marketplace. In the ramp-up to Y2K many organizations focused on cleaning up and stabilizing existing software while at the same time creating new defect free software using more comprehensive development processes, attempting to avoid any problems when entering the New Year. Since 2000 many companies have moved to outsourcing situations which, through contractual requirements, focus on very high quality software, which can have a negative impact on developer productivity. In addition to a shift towards outsourcing and higher quality software, the Carnegie Mellon SEI/CMM methodology has become very widely accepted across the industry. As organizations mature through the CMM there is a requirement for improved processes and documentation, all of which leads to higher quality software. For many organizations, this requirement adds to the overhead associated with delivering software due to the additional activities performed and can result in a negative impact on productivity. We discuss the implications of this productivity decline in the context of making comparisons with our preliminary hedonic price indexes, below.

Hedonic Price Indexes

It is well understood that matched-model price index techniques fail to fully account for quality change. As a result, matched-model price indexes typically understate (overstate) true price declines (increases). Therefore, when possible, it is preferable to develop a price index that explicitly adjusts for changes in product quality. The second price index approach we propose to implement will attempt to adjust for quality change by estimating a hedonic regression equation. In this analysis, the individual software project will be the unit of observation. Such an approach allows for the inclusion of projects that are omitted from a matched-model approach due to non-matching, and allows one to utilize the variation across *all* products to more precisely account for quality change. This feature is particularly important given the heterogeneity that exists among custom software projects, and as noted above, overcomes the problem of the severe gaps in coverage in the data that makes the matching of elementary units not feasible in all periods. In addition, we are able to control for a number of potentially relevant price determining factors that are not incorporated in the matched-model price indexes discussed earlier. Below is a description of the baseline hedonic regression model we have developed for custom and own-account software, as well as several extensions to this model.

Econometric Specification

We employ the same general semi-log regression specification that has been used in the literature on hedonic price indexes.[11] Unlike the matched-model price indexes discussed above, here the dependent variable is the natural logarithm of the price (i.e., it is not the price per adjusted function point as in the matched-model specifications). Instead, in our Base Case analyses, we control for project size by including such measures as explanatory variables in the hedonic price regressions. For some projects, the price will represent the contracted price, while for other projects the price will be computed using wage rates and information on project hours. Such a calculation is common within the custom software industry to price and value a software project because the primary cost/price driver is the amount of project labor hours, and thus, we are confident that this approach yields the most reliable measure of price available. In addition to the price determining attribute variables described below, annual time dummy variables are added as explanatory variables to allow for the calculation of a quality-adjusted price index. Our Base Case regression takes the form:

$$\ln p = \alpha + \beta_i X_i + B_y D_y + \varepsilon \qquad (1)$$

where:

p	= Price for a given project in a given year;
β_i	= Regression coefficients for i price determining attributes, $i = 1, \ldots n$;
β_y	= Regression coefficients for 10 annual dummy variables;
α	= Intercept term, representing reference case; and
ε	= Independently distributed random disturbance term.

We refer to equation (1) as our Base Case hedonic regression model.

[11] See Berndt [1991, Ch. 4].

Price Determining Attribute Variables

In general, we believe the variation in custom and own-account software prices can be explained by several price determining attributes. Some variables that we expect to impact the price of software vary across projects and time, but are not necessarily related to pure quality change. These variables are primarily the result of changes in the software development process that have been driven by software development and business organizations that have demanded higher quality software and timely delivery. Other variables are more explicitly related to quality. Below is a more detailed description of the price determining attributes we include as explanatory variables in our hedonic analyses. Many of these variables have been explained in more detail in Part 2 above.

Discussion of Explanatory Variables:

We have identified a number of potential price determining attribute variables that are available in the Q/P database for constructing our hedonic regression models, including:

- Value Adjustment Factor
- Project Size
- Productivity
- Project Type
- Software Type
- CMM Level
- Platform
- Industry

From a software user's perspective, one measure of quality of custom and own-account software projects is the Value Adjustment Factor (VAF). Therefore, we include this variable in our Base Case hedonic regressions as a continuous variable, ranging from .65

to 1.35. *A priori*, we expect a positive relationship between the VAF and price (i.e., higher quality projects are typically associated with higher prices).

Project size, which is measured by the number of function points for each project, is another price determining factor for custom and own-account software projects. In our Base Case specifications, this variable may enter the regression equation as either a continuous variable ranging from 3 to 22,000 or as a set of dummy variables organized by project size, as discussed in the matched-model price index approach.[12] *A priori*, we expect a positive relationship between project price and function point count (i.e., larger projects are typically more expensive).

Productivity, which is measured by the number of adjusted function points produced per hour for each project, can also impact the price of custom and own-account software. In our Base Case specification, this variable is treated as a continuous variable ranging from 0.01 to 8.39. *A priori*, we expect a negative relationship between project price and productivity (i.e., increases in productivity result in lower prices/project costs). The inclusion of this variable as an explanatory variable in our hedonic regression models warrants further discussion since including what may be interpreted as a supply-side variable departs from the traditional hedonics literature. Because the objective of this research project is to construct price indexes that account for changes in quality *and* productivity, the inclusion of such a variable is warranted. Moreover, since some of the productivity decline observed in this industry has come about in an effort to increase quality, we believe this variable serves as a proxy for otherwise unmeasured quality change in custom and own-account software. Furthermore, the custom and own-account software is highly labor-intensive. As a result, we believe it is critically important to include this variable in our hedonic regression models.[13]

---

[12] Because adjusted function points are the product of unadjusted function points and the Value Adjustment Factor, we use counts of unadjusted function points, rather than adjusted function point counts, when project size is measured as a continuous variable. This allows for separate measurement of quality, as measured by the Value Adjustment Factor, and project size.

[13] Because this explanatory variable and the dependent variable are a function of hours, a potential statistical problem may arise. Borjas [1980] finds that the appearance of hours on both sides

We also include a dummy variable in our Base Case hedonic specification to control for Project Type. This variable takes on a value of 0 if it is a new development project and a value of 1 if it is an enhancement to existing software. *A priori*, we expect this variable to have a negative effect on price, as constructed, since enhancement projects are typically less expensive than new development projects, all else equal.

The variable Software Type identifies an individual project as either a custom software project developed by a commercial developer (on contract or outsourced) or an own-account software project developed in-house by a company. To account for this price determining factor, we construct dummy variables for such projects, and use Custom-Contract projects as the base value. All else equal, Custom-Contract prices tend to be on average higher than the prices of all other project types and therefore one needs to control for this variable. Thus, one needs to control for this project attribute.

Another project factor that should be considered is the level of maturity of the processes utilized to develop the software and manage the project, which can be measured by the CMM level of a software organization. The level of process maturity can impact both the price and value of software. Commercial software development organizations that are assessed at high CMM levels can often charge a premium for their services. The customer can benefit as well by receiving software that is of higher quality (fewer defects) and is less costly to maintain due to improved documentation and standards for development. For this reason, the CMM level should be considered as a potential project factor independent variable in a hedonic regression. In our base case specification, CMM level has been incorporated as a set of dummy variables given the non-monotonic relationship that we expect between price and CMM level, all else equal.

The Platform of the hardware/software environment may impact the productivity of the developer and therefore needs to be considered as a price determining factor. As such,

downward biases the estimates as long as there are errors of measurement in the observed measures of labor supply.

we include a set of dummy variables in our Base Case Specification to account for the hardware/software environment of a custom or own-account software project.

In our Base Case specification, we do not use the industry groupings described in the matched-model price index section, rather we are able to use the 12 industry categories available in the updated Q/P Management data given that matching is not an issue for the hedonic price index. Since it is possible that different industries have different software needs and different levels of productivity in developing custom or own-account software, it is preferable to treat each industry separately in specifying our hedonic equation. As such, Client Industry has been incorporated in our Base Case specification as a set of dummy variables.

As a final control variable, we include dummy variables to account for differences in the way function points have been counted over time, or the CPM Version. The International Function Point Users Group is the organization responsible for defining and maintaining the rules for how function points are counted, which are documented in the CPM. The benchmark database contains data representing three (3) versions of the CPM (3.4, 4.0 and 4.1). Most major releases of the counting practices manual are accompanied by a conversion factor representing the impact the rule changes have on function point counts created in the most recent previous version. The percent change in function point counts from CPM 3.4 to 4.0 is approximately -20 percent. The percent change in function point counts from CPM 4.0 to CPM 4.1 is -3.24 percent. The resulting percent change from one CPM version to another will impact the size of the project and the resulting productivity and cost per function point, which is why it needs to be accounted for in the analysis.

Results:
As described above, our Base Case regression controls for project type, software type, CPM version, CMM level, platform, industry, productivity, project size, and value adjustment factor. The dependant variable is the price of each custom or own-account software project, and each observation in the regression is an individual software project.

Because we limit the data to the 1993 to 2003 time period, our hedonic analyses are based on 5,419 observations. When computing the resulting hedonic price indexes discussed below, we implement two-year moving averages for the index values. As a consequence, our hedonic price indexes cover the time period 1994 to 2003. We implement this smoothing technique primarily because the price measure in our data reflect the final price paid for the custom software project even though a project may span multiple months or years, often between one and two years. Moreover, such a technique may help reduce year-to-year volatility that arises from the changing nature of the sample (previously discussed), which may not be fully representative of all custom and own-account software projects. Using a two-year moving average provides these benefits of smoothing, without loosing multiple years of data.

As seen in column A of **Appendix D**, the parameter estimates on many of the price determining attribute variables are statistically significant and of the expected sign. For example, the coefficient for the VAF is positive and significant at the 1 percent level, indicating that price of custom and own-account software projects increases with quality. Also, project size, measured as a continuous variable, is positive and significant at the 1 percent level. In addition, the coefficient on productivity is negative and significant at the 1 percent level, as expected. The explanatory variables used in this Base Case specification explain approximately 64 percent of the variation in custom and own-account software prices, as indicated by the R-squared measures that are reported.

As shown in **Exhibit 10A**, over the 1994 to 2003 time period, prices of custom and own-account software projects rose at an annual rate of 11.7 percent based on traditional hedonic price indexes. The price changes for the two sub-periods 1994 to 1998, and 1998 to 2003, are -12.7 percent and 36.0 percent, respectively, indicating that price increases accelerated in the latter half of the sample, consistent with the findings for the matched-model price index. Note that the annual price increase of 11.7 percent measured here is lower than the 26.2 and 12.6 percent increase for the average price and average price per adjusted function point, respectively, as shown in **Exhibits 7A** and **7B**.

Within the context of traditional hedonic price indexes, the issue of parameter stability has received some attention in the literature, particularly in the context of products subject to rapid technological change.[14] From a conceptual standpoint, the effect of certain price determining factors may not be constant over time if the composition of products sold and their underlying attributes change frequently, which is a common feature in information technology industries. Further, as noted above, the data on which this analysis is based show considerable year-to-year heterogeneity along a number of dimensions. Taken together, these factors suggest that the assumption of inter-temporal parameter stability may not hold for custom and own-account software.[15] One approach that has been suggested to overcome this problem is the construction of chained biennial hedonic price indexes.[16] This approach is less restrictive than the traditional approach in that it allows for more flexibility in parameter values over time.

We implemented this approach, and found very different results than those suggested by the traditional matched-model or hedonic price indexes. We estimated sets of ten year-by-year regressions for each hedonic specification, where each biennial regression covering adjacent years reflected software projects and project attributes relevant for the sets of adjacent years. As shown in **Appendix E**, in general, the parameter estimates for many of the price determining attribute variables remain statistically significant and of the expected sign for the biennial regressions, although the estimates can vary somewhat year to year. This is particularly true for the VAF variable, which does not remain significant in all biennial time periods. As expected, however, parameter estimates on project size are consistently positive and highly significant while parameter estimates on productivity are consistently negative and highly significant. On average, the biennial regressions explain approximately 63 percent of the variation in price, although again

[14] See, for example, Triplett [2004] and Pakes [2003].

[15] As one test of this assumption, we introduced interaction terms into our Base Case regression specification to allow for the possibility that the effect on price of project size, the value adjustment factor, and productivity may vary by year. In general, many of these interaction terms were significant at the .01 level, which strongly rejects the assumption of inter-temporal parameter stability.

[16] See, for example, Berndt and Rappaport [2002]. Indeed, according to Moylan and Robinson [2003], the BEA has recently adopted this approach to construct price indexes for products subject to significant change over time.

there is considerable year-to-year variation in the R-squared measure. As shown in **Exhibit 10A**, over the 1993 to 2003 time period, prices of custom and own-account software projects declined at an annual rate of 19.4 percent, which is more in line with the findings of Ethiraj, Kale, Krishnan, and Singh [2004]. The price changes for the two sub-periods 1994 to 1998, and 1998 to 2003, are -25.5 percent and -14.1 percent, respectively, which is quite different from the results of the traditional hedonic regressions discussed above. As shown in **Exhibits 10B-10D**, this general pattern holds across all hedonic regression specifications.

As an alternative Base Case specification, we re-estimated the model, dropping the unadjusted function point as an explanatory variable of price, and adding in project size categories based on function point counts. The results of this regression are reported in column B of **Appendix D**. The resulting coefficient estimates were largely unchanged in terms of sign and significance, although the positive coefficient on the value adjustment factor was no longer significant at the 1 or 5 percent levels of significance, and the R-squared increased from approximately 0.64 to approximately 0.81. The relationship between project size and price is monotonic, i.e., increases in project size based on function point count groupings are associated with higher prices, and all coefficients on the variables are significant at the 1 percent level. As shown in **Exhibit 10B**, the resulting CAGR decreased from 11.7 to 4.9 percent, although a similar acceleration in price increases was observed in the latter half of the sample. As before, we also estimated a series of biennial regressions described above, which are reported in **Appendix E**. When the resulting price changes were chained together, the resulting price decline was 4.4 percent annually over the 1994 to 2003 time period.

In view of the fact that custom and own-account software projects are just that, i.e., customized to meet the software needs of individual firms/organizations, we accounted for this potential wide variation in heterogeneity across software projects by including client fixed effects variables. That is, dummy variables for each unique client were added as explanatory variables in the two hedonic specifications previously described in place of CMM levels, platforms, and industries since we do not expect much within-client variation across such measures. The results from these Client Fixed Effects models are reported in column C and D of **Appendix D**. The overall fit of the Client Fixed Effects regressions is slightly higher than the Base Case, explaining approximately 68 and 82 percent of the variation in price, respectively. As shown in **Exhibit 10C** and **Exhibit 10D**, the resulting CAGRs, which correspond to the Base Cases of A and B, are 0.1 and 4.8 percent, respectively. The corresponding CAGRs for the chained biennial hedonic price indexes are -16.0 and -8.0 percent.

It is worth noting that the resulting hedonic price indexes are quite different depending on whether project size is measured in levels or by creating project size groups based on the number of function points. For example, in our base case regressions (Exhibits 10A and 10B) using levels yields an 11.7 percent growth rate for the traditional hedonics compared to 4.9 percent when project size is measured in groups. In contrast, in our client fixed effects regressions (Exhibits 10C and 10D), using levels yields an annual decline of 16.0 percent for the chained-biennial hedonics compared to a decline of 8.0 percent when project size is measured in groups. Thus, while different, no systematic pattern appears to exist. The observed differences may indicate that a non-linear relationship may exist between function points and project price. However, it may also result from the specific project size groupings we have used for this analysis. Therefore, we believe additional research into the relationship between project size and price appears to be warranted.

Comparison of Preliminary Price Indexes to Government Price Indexes

In **Exhibit 11**, the BEA price indexes for Custom and Business Own Account software are compared to our chained biennial hedonic price indexes, over the common time period for which the published BEA indexes overlapped with our data, i.e., 1997 to 2003. The BEA indexes display almost no price change over the relevant time period. Recall that the BEA price indexes are a weighted average of the BLS prepackaged software price index and an input cost index (based on an assumption of zero labor productivity change). Given that neither index has shown much price movement over this time period, it is not surprising that the official BEA deflators show price increases of only 1.3

and 1.4 percent, annually, for Custom and Business Own Account Software. By contrast, the chained biennial price indexes that control for changes in quality and productivity and allow for changes in the composition of the data over time show price changes ranging from -14.7 percent to 10.6 percent over the same time period.

As a check on the robustness of the results of the Base Case and Client Fixed Effects hedonic regression models with price as the dependent variable reported in **Appendix D** and **Appendix E**, we developed a set of sensitivity analyses that are reported in **Appendix F** and **Appendix G**.[17] In doing so, we re-estimated our Base Case and Client Fixed Effects models using price per adjusted function point, rather than price, as the dependent variable. Note that this approach is similar to that of the matched-model price index where price per adjusted function point is calculated at the elementary unit level. Because our dependent variable is price per adjusted function point, we excluded the VAF and continuous measure of function point counts from these analyses as explanatory variables in a hedonic regression. Overall, the CAGRs resulting from these alternative traditional hedonic price indexes range from 4.8 percent to 9.5 percent, compared with the range of -3.6 to 10.3 percent for the set of indexes where price is the dependant variable. Thus, the overall range of price change across all traditional hedonic specifications for the 1993 to 2003 time period is -3.6 to 10.3 percent per annum.[18] Similarly, the CAGRs resulting from the alternative chained biennial hedonic price indexes range from 3.7 percent to 5.0 percent, compared with the range of -20.0 to -6.1 percent for the set of indexes where price is the dependant variable. Thus, the overall range of price change across all chained biennial hedonic specifications for the 1993 to 2003 time period is -20.0 to 5.0 percent per annum.

Additional analyses were conducted on subsets of the benchmark data. One analysis involved the function point counts for approximately 300 projects, spanning four years from four companies to determine if the contribution of any one of the five function types

[17] Note that these results are not report as two-year moving averages.

[18] A number of alternative econometric specifications were estimated in developing our preliminary hedonic price indexes. In all cases, the regression results and corresponding price index were qualitatively similar to those reported above.

(ILF, EIF, EI, EO or EQ) influenced the productivity rates or price of a project more than the others. The ratios of the function point count to each function type were compared across the complete dataset to determine on average how each function type contributed to the size of the function point count. A similar analysis was then completed against the projects that were in the top quartile of highest productivity (lowest cost). This analysis resulted in very similar ratios for the 5 function types when compared to the complete dataset. This indicates that specific function types do not influence the productivity or cost of projects.

A separate analysis was conducted on 2,212 projects spanning all years from 14 companies to determine if project duration (number of calendar days required to complete a project) influenced the price of the project. The analysis indicated that a shortened or lengthened project schedule does not consistently influence the price of the project.

Part 4 – Proposed Price Indexes: Recommendations and Implications

As discussed above in Part 3, we have developed a number of alternative price indexes for custom and own-account software using the Q/P Management database that was developed as part of this project. While traditional matched-model approaches are commonly relied upon by the BEA and BLS, such indexes were difficult to implement for custom and own-account software. In particular, the changing product/industry/platform/client mix observed in the data made it difficult to find matches over time for appropriately defined elementary units. In addition, while the traditional hedonic approach did overcome some of the limitations present in the matched-model price index approach, this approach ultimately suffered from the year-to-year volatility of the custom and own-account software data coverage. Specifically, the assumption of inter-temporal parameter stability, which has recently received attention in the hedonic price index literature, was shown not to hold. As a result, given the rate of turnover in the sample in terms of industry and project type coverage, we recommend adopting a biennial regression hedonic framework that appropriately allows for the variety of changing attributes and clients in this industry in the custom and own-account database. Moreover, because custom and own-account software can be quite unique, we believe it is preferable to include client fixed effects in such regression models. However, caution must be used in interpreting the chained biennial hedonic price indexes since it was not always possible to use the exact same regression model over time due to changes in the data coverage. In addition, the parameter estimates on the time dummy variables were not always significant.

The overall goal in estimating custom and own-account software price indexes is to develop appropriate deflators for nominal private fixed investment in custom and own-account software in the National Income Accounts. To assess the impact of our proposed price indexes on the measurement of *real* investment in custom and own-account software, we show the results of deflation by our various proposed indexes in **Exhibit 12**. Over the 1994 to 2003 time period nominal private investment in custom software rose from \$21.4 billion to \$47.0 billion, an annual increase of 9.1 percent. Over the same period, investment in own-account software rose from \$27.0 billion in 1994 to \$71.1 billion in 2003, or an annual increase of 11.5 percent. Adjusting the measures of nominal investment by our four sets of biennial price indexes results in annual increases in *real* private investment in custom software in the range of 14.2 to 35.3 percent, and 16.6 to 38.2 percent per annum in real increases in private investment in own account software. Given that our preliminary preferred price indexes show price declines relative to the current official indexes used by BEA, current BEA deflators may understate true real investment in current and own-account software. However, this conclusion is in part driven by the sharper decline in prices between 1994 and 1995. If these periods are excluded from the analysis or if one adopts a more extreme type of moving average in computing price changes for custom and own account software over time, it is likely that the proposed price indexes will show less of a price decline, and that real investment in current and own-account software would be correspondingly lower.

References

Abel, Jaison R., Ernst R. Berndt, and Alan G. White [2003], "Price Indexes for Microsoft's Personal Computer Software Products," Cambridge, MA: National Bureau of Economic Research, Working Paper No. 9966, September.

Berndt, Ernst R., *The Practice of Econometrics: Classic and Contemporary*, Reading, MA: Addison-Wesley Publishing Company, 1991, Ch. 4.

Berndt, Ernst R. and Neal J. Rappaport [2003], "Hedonics for Personal Computers: A Reexamination of Selected Econometric Issues," MIT Sloan School of Management, Draft Manuscript, August.

Borjas, George [1980], "The Relationship Between Wages and Weekly Hours of Work: The Role of Division Bias," *Journal of Human Resources*, 15, pp. 409-423.

Brynjolfsson, Erik and Chris F. Kemerer [1996], "Network Externalities in Microcomputer Software: An Econometric Analysis of the Spreadsheet Market," Management Science, 42, pp. 1627-1647.

Ethiraj, Sendil K., Prashant Kale, M.S. Krishnan, and Jitendra V. Singh [2004], "Determinants of Price in Custom Software: A Hedonic Analysis of Offshore Development Projects," Unpublished Mimeo, University of Michigan Business School, March.

Gandal, Neil [1994], "Hedonic Price Indexes for Spreadsheets and an Empirical Test of Network Externalities," RAND Journal of Economics, 25, pp. 160-170.

Gandal, Neil [1995], "Competing Compatibility Standards and Network Externalities in the PC Software Market," The Review of Economics and Statistics, 77, pp. 599-608.

Grimm, Bruce and Robert Parker [2000], "Software Prices and Real Output: Recent Developments at the Bureau of Economic Analysis," paper presented at the National Bureau of Economic Research, Program on Technological Change and Productivity Measurement, Cambridge, MA, March 17.

Grohn, Andreas [n.d.], "Network Effects in PC Software: An Empirical Analysis," unpublished manuscript, Kiel, Germany: Kiel University.

Harhoff, Dietmar and Dietmar Moch [1997], "Price Indexes for PC Database Software and the Value of Code Compatibility," Research Policy, 26, pp. 509-520.

McCahill, Robert J. [1997], "An Hedonic Study of Prepackaged Software," Master's Thesis, Department of Economics, Virginia Polytechnic Institute and State University, Blacksburg, VA.

McKinsey Global Institute [2001], "United States Productivity Growth, 1995-2000," Washington DC: McKinsey Global Institute, October.

Moylan, Carol E. and Brooks B. Robinson [2003], "Preview of the 2003 Comprehensive Revision of the National Income and Product Accounts," *Survey of Current Business*, 83 (9), pp. 17-32.

OECD, "Report of the OECD Task Force on Software Measurement in the National Accounts," 2002.

Oliner, Stephen D. and Daniel E. Sichel [1994], "Computers and Output Growth Revisited: How Big is the Puzzle?" Brookings Papers on Economic Activity, 2, pp. 273-330.

Pakes, Ariel [2003], "A Reconsideration of Hedonic Price Indexes with an Application to PC's," American Economic Review, 93(5), December, pp. 1578-1596.

Prud-homme, Marc and Kam Yu [2002], "A Price Index for (Prepackaged) Computer Software Using Scanner Data," draft manuscript, Ottawa: Statistics Canada, Prices Division, July.

Triplett, Jack E. [2004], “Handbook on Hedonic Indexes and Quality Adjustments in Price Indexes: Special Application to Information Technology Products,” Directorate for Science, Technology and Industry, Paris, OECD, June.

Wasshausen, David [2003a], “2003 Benchmark Revision: Proposal to Improve the Price Index for Private Fixed Investment in Own-Account Software,” memo to Brent Moulton, February 26, 2003.

Wasshausen, David [2003b], “Follow-up: 2003 Benchmark Revision: Proposal to Improve the Price Index for Private Fixed Investment in Own-Account Software (B-57),” memo to Brent Moulton, May 15, 2003.

White, Alan G., Jaison R. Abel, Ernst R. Berndt, and Cory W. Monroe [2004], “Hedonic Price Indexes for Personal Computer Operating Systems and Productivity Suites,” Cambridge, MA: National Bureau of Economic Research, Working Paper No. 10427, April.