Distr.
GENERAL

Working Paper No.15
29 April 2009

ENGLISH ONLY

UNITED NATIONS STATISTICAL COMMISSION and ECONOMIC COMMISSION FOR EUROPE CONFERENCE OF EUROPEAN STATISTICIANS

EUROPEAN COMMISSION STATISTICAL OFFICE OF THE EUROPEAN COMMUNITIES (EUROSTAT)

ORGANISATION FOR ECONOMIC COOPERATION AND DEVELOPMENT (OECD) STATISTICS DIRECTORATE

Meeting on the Management of Statistical Information Systems (MSIS 2009)
(Oslo, Norway, 18-20 May 2009)

Topic (ii): Governance

USING A SOFTWARE DEVELOPMENT PROCESS TO IMPROVE THE 2007 U. S. ECONOMIC CENSUS

Supporting Paper

Prepared by Barry F. Sessamen, U. S. Census Bureau[1]

I. INTRODUCTION

1. The U.S. Census Bureau is the largest statistical agency in the United States federal government. It serves as the leading source of quality data about the nation's people and economy. The Census Bureau achieves that mission by producing high quality, relevant statistical information as defined by the U.S. Office of Management and Budget's (OMB) guidelines for quality, objectivity, utility, and integrity of information. The OMB standards and guidelines dictate that Federal statistical organizations design surveys and censuses to efficiently produce reliable data, use methods that are documented, and present results in a manner that makes the data as accessible and useful as possible.

II. U.S CENSUS BUREAU

2. The Census Bureau is organized into three major subject matter areas: (1) the Decennial Census Directorate, responsible for the 10 year population counts, (2) the Economic Programs Directorate, responsible for the measures of the U.S economy and (3) the Demographic Directorate, responsible for Census Bureau programs on the U. S. population and its characteristics, income and poverty, and housing. There are also various specialized staffs across the Census Bureau devoted to methodology, statistical research, information technology infrastructure and administrative support services.

[1] Barry F. Sessamen, U. S. Census Bureau, 4600 Silver Hill Road, Room 7K105, Washington, D.C. 20233, USA (barry.f.sessamen@census.gov)

III. THE ECONOMIC PROGRAMS DIRECTORATE

3. The Census Bureau's Economic Programs Directorate is responsible for statistical programs that measure and profile businesses and government organizations in the United States. Its activities are wide reaching and include (i) the conduct of an Economic Census and a Census of Governments every five years; (ii) the conduct of over 100 current surveys taken monthly, quarterly, and annually, including twelve principle economic indicators for the U.S economy; (iii) the compilation of administrative records from other government agencies; and (iv) the conduct of numerous research and technical studies. In addition to these activities, the Economic Directorate also conducts a large number of reimbursable studies and joint projects for other U.S federal agencies that leverage the Directorate's processing infrastructure and core competencies. In order to manage the vast variety of programs that the Economic Directorate undertakes, there are nine organizations that bring specific competencies and expertise to each program. The five subject matter expert divisions specialize in various aspects and industries of the economy and work in tandem with the Information Technology and Programming areas, the coordination and planning area, the statistical methods and research area, and the Center for Economic Studies in order to collect, process, and disseminate data.

4. In a constantly changing and increasingly complex economy, the Economic Programs Directorate has positioned itself to respond to these changes while providing the most efficient use of its resources and expertise. These challenges include:

- Measuring a constantly changing economy
- Improving respondent cooperation and maintaining high quality data
- Improving programs in the face of constrained resources
- Attracting, developing, and retaining a skilled workforce
- Protecting data and maintaining data security

5. The Census Bureau must address these challenges successfully in order to further the mission, strategic goals and objectives of the U. S. Department of Commerce (DOC), its parent organization. In order to meet that need, the Directorate has two strategic goals directly related to improving the quality of its programs. One goal is to "Increase the quality of economic statistics by improving the relevance, accuracy, and timeliness". The other goal is to "Operate more efficiently." These goals are supported by multiple objectives that include focusing on building and maintaining a strong IT Software Development program, one based on sound project management practices as well as proven IT and business practices.

6. The economic census serves as the cornerstone of the programs conducted by the Economic Programs Directorate. It is a complete enumeration of some 25 million business establishments in the 50 states and the District of Columbia, Puerto Rico, and Island Areas that include the U.S. Virgin Islands, Guam, the Northern Mariana Islands, and American Samoa. Title 13, United States Code, Section 131 directs the U.S. Census Bureau to take an economic census every 5 years, covering years ending in "2" and "7." Further, the United States Congress appropriates funds for this purpose as part of the U.S. Census Bureau's budget. Among its most important roles is serving as the basis for the Business Register (BR), the complete listing of all business establishments in the United States. The BR serves as the mailing list for the economic census as well as the sampling frame for the economic surveys conducted by the directorate. Thus, the quality of all economic area programs begins with the quality of each economic census.

IV. THE ECONOMIC CENSUS

7. The economic census covers the Retail, Wholesale, Service, Finance, Insurance, Real Estate, Utilities, Transportation, Manufacturing, Mining, and Construction sectors of the economy. Cutting across all of these sectors are a broad range of activities, including the following:

(i) Planning, managing, and administering the program including the application of specific project management methods and tools to the program.

(ii) Determining the content of public data products and the collection upon which they are based.

(iii) Promoting high response rates and early reporting through our Customer Relationship Management Program.

(iv) Developing requirements for instrument design, data capture, microdata editing/cleansing, workflow and data access, management information systems, problem resolution, tabulation, macrodata analysis, disclosure avoidance, data product review/approval, quality assurance, dissemination, and other processes needed to conduct an effective economic census.

(v) Designing, developing, testing, and implementing software systems to meet user requirements while ensuring compliance with IT security mandates and data stewardship.

(vi) Determining software development procedures and standards, including project management, and instituting software quality assurance mechanisms for independent monitoring and enforcing process adherence.

(vii) Implementing guidelines for ensuring and maximizing the quality, objectivity, utility, and integrity of economic census statistical information.

(viii) Carrying out production activities for census data collection, processing, analysis, product preparation, and dissemination.

(ix) Closing out the program, evaluating performance, and documenting lessons learned.

8. In order to accomplish a successful economic census, it is necessary to master this wide variety of activities that involve statistical methods, software development processes, and project management.

V. THE SOFTWARE DEVELOPMENT PROCESS FOR THE ECONOMIC CENSUS

A. Background

9. As we reflected on the challenges and successes of the 2002 Economic Census, we discovered that many resources were spent building and testing software that collected, processed, and disseminated the data. For the 2007 Economic Census, the triple constraint of scope, timeline, and resources would likely be even more constrained. Expectations were for scope to increase, timelines to become shorter, and resources to remain fixed or decline. It was the typical ‘do more with less’ scenario.

10. In order to remain successful in this environment, we needed to create a software engineering culture that would introduce more structure, control, and accountability into the Economic Directorate’s software projects, thus improving software product quality and efficiency.

11. For the 2007 Economic Census, we hoped to achieve our objectives by improving our software development and the supporting project management by implementing for the first time a defined and repeatable Software Development Process (SDP).

B. Defining the Software Development Process

12. Software development projects present a special challenge to project managers and thus there is a unique set of software development project management methodologies that are considered best practices. We evaluated several methodologies such as the Capability Maturity Model (CMM), the Team Software Process (TSP), and others, and took from them the most relevant and helpful practices based on our particular situation. We documented both a minimum and a recommended set of activities. The minimum process defined only the essential components that would be standard for all software development. The recommended process defined software development components of a more extensive nature. This dual concept allowed certain projects to start slowly by using the minimum process, but allowed individual teams to implement processes from the more extensive recommended list whenever beneficial to team objectives. Thus, it would mitigate the cultural change by starting slowly, but would support continual process improvement over time. The defined minimal process would eventually expand. It would also allow teams to tailor the software development to fit the customer’s specific needs.

13. The initial implementation was to focus on strengthening areas of weakness and/or changes that would provide the largest benefit. The overriding goal was to ensure that our products are on time, within budget and cover the agreed upon scope.

14. The Software Engineering Institute (SEI) had long established volumes of data supporting their conclusion that the more time spent in the 'up-front' processes such as the elicitation and analysis of requirements, the system design, and the early inspection of code, the less re-work would be needed later in the process when the costs of changes to these activities would be prohibitive. The time and resources saved would allow us to expend them on more value added activities such as data analysis and review. We incorporated these principles into our defined process.

15. The Recommended Software Development Process is depicted in Attachment A. It is a complete methodology based on software development best practices and their role in our organization. The process can be broken down into the following main components:

- Plan - Project Management Planning and Tracking
- Requirements
- Design
- Program Coding
- Testing
- Ongoing Activities

These components and their key subcomponents will be further defined in sub-sections C. to H.

C. Plan - Project Management Planning and Tracking

C.1 Project Charter

16. The first objective in the planning stage is to ensure that there is merit to the activities being planned and to establish the framework in which it can be successful. All new projects would begin by creating a Project Charter. The Charter defines the purpose of the project, what is in-scope and out-of-scope, the resources being assigned and the major milestones that must be met. The Charter should be no more than one or two pages long. Once accepted, the project manager will need to coordinate with the support manager and logistics staff to register the project and secure the necessary IT resources such as software tools, hardware, and disk storage, and to ensure that there are no security issues or concerns.

C.2 Project Plan

17. Once the team has a Charter for a new project, the project manager, with the assistance of the team, will create a Project Plan. The Project Plan begins to identify not only what will be done, but how it will be conducted. It should minimally include:

- A User Concept Diagram, which can be a brief sketch of how the user envisions the project,
- Interrelated Projects and how they will interact
- Deliverables
- The expected Software Life Cycle design
- Roles and Responsibilities for each team member
- Full Time Employee (FTE) estimates depicting how many resources will be involved, from what organization, and to what extent
- A High Level Schedule
- A Risk Management Plan that identifies major risks and mitigation strategies for those risks
- A Change Control Plan for communicating, implementing and tracking approved changes
- The Communication Plan depicting not only how information will be dispersed and team meetings conducted, but how issues will be charted and decisions made and documented

C.3 Project Schedule

18. The Project Schedule should be produced from a breakdown of the project into small tasks. Each task should have a single person assigned as being responsible for its completion. The schedule should identify any dependencies between tasks. For the Economic Census, the individual project schedules rolled-up to a comprehensive master schedule. Both are reviewed on a periodic basis, and discrepancies between planned and actual dates addressed.

D. Requirements

19. Projects will define the system requirements separately from the design.

D.1 Requirements Elicitation

20. Requirements Elicitation is the process for discovering a capability or condition needed by a user or to solve a problem or achieve an objective. It is achieved generally through dialogue with users/stakeholders and examination of the environment. Typical methods for eliciting requirements include interviews, scenarios (use cases), prototypes, facilitated meetings, observation, and others.

D.2 System Functional Requirements

21. System Functional Requirements are the elicited capabilities stated in a user's language that define ***what*** the software system or component must do. Functional requirements can be further divided into high-level requirements and detail requirements. High-level requirements clearly identify all of the distinct capabilities that a software system or component must provide in terms of inputs, behavior, and results. This can be calculations, technical components, data manipulations and processing, or other system actions. Detail requirements decompose the high-level requirements to the level of detail needed for complete specifications of each capability –i.e., detail sufficient for computer program design and coding.

D.3 System Test Plan

22. Once System Requirements are defined, a System Test Plan should also be developed. It cannot be conducted until the system is actually defined and coded, but this early rendering of the test plan will help ensure that testing can be traced back to requirements, and can commence as soon as the testing phase is reached. The Test Plan will document the scope of testing, the resources assigned, the test environment, and procedures for reporting errors and defects. A limited set of test cases depicting inputs and expected results can also be defined, although it may need to be enhanced later in the process.

D.4 System Non-Functional Requirements

23. System Non-Functional Requirements are conditions or constraints on the project due to the availability of hardware, software, operating systems or any other physical conditions. They generally describe a quality of service characteristic such as performance, availability, reliability, scalability, maintainability, portability, security, cost, and others. Because of the large user base for the Economic Census, we decided to focus on the performance of our interactive web-based systems, and created a specific test plan for that purpose.

D.5 Requirements Review and Inspection

24. A software solution that does not meet the requirements will not meet the needs of the users. It is critical that the requirements documents and test plan be reviewed and inspected by all of the involved users and business analysts. A review of the requirements will provide feedback about possible omissions, errors, or other problems, and ensure completeness. An inspection is a more rigorous activity in which a set of requirements is evaluated against standards and constraints of the process such as being unambiguous, complete, consistent, prioritized, verifiable, and traceable. Violations are noted as defects.

E. Design

E.1 High Level Design

25. The High Level Design is the process of analyzing design alternatives and defining the components and interactions among those components. It is basically the framework or blueprint that depicts the various relationships between components such as the clients and servers, databases, filters, pipes, and layers of a hierarchical system. Additionally, the framework should include the pathways of interaction between the components, such as communication links and protocols, information flows, and access to shared storage. These components must then be mapped to a hardware view. This can be extended to individual programs or modules. An integration test plan can then be created for system and integration testing.

26. Once all of the operational scenarios connecting programs to database are complete, an entity-relationship diagram can be created to serve as the basis for the physical design of the database.
Reviews and inspections of the High-Level Design should be checked against the requirements to ensure that all are met and that no actions are taken unless they are part of the requirements.

E.2 Detail Level Design

27. The Detail Design is the lowest design level. It identifies the logic specification required for the program developer to code the program or module.
The Unit Test Plan involves testing the smallest possible unit of an application. It is the first chance to test pieces of a system or application to ensure that they function as planned. It defines the test environment, the unit test cases to be run and the expected results.

F. Program Coding

28. Coding is the process by which a software developer translates the solution for requirements into a given computer language and its associated syntax. The process for coding computer programs provides a checklist to identify the coding standards and practices that should be followed to produce a more uniform product across the program area. Code standards make it easier to integrate modules from different programmers or even from different languages, into the same program. Developers are encouraged to perform a Personal Code Review to document their code defects so that they become more aware of personal coding issues, which they can improve in the future. Code walkthroughs and inspections familiarize a larger audience with the program functionality, providing better support coverage and an opportunity for all to learn from each other's code. The review team should compare the code to the specifications on which it was based, as well as the coding standards. Since the programming standards will be enforced in inspections, it is important that the standards be short – less than two pages. It is also the last chance to find and remove defects before testing, which usually takes a lot less time to solve then defects found during or after testing, so the process should be rigorous. It is up to the code author to answer all questions and resolve any problems.

G. Testing

29. The test process is the last chance to find and fix any defects and omissions from requirements to design to code. There are three types of testing: Unit Tests, System Tests (including Integration, Regression, and Performance), and User Acceptance Testing. Testing is conducted from the bottom up – i.e., from the smallest components to the largest. Program modules are tested separately in Unit Tests, then integrated and tested as a subsystem, then all subsystems integrated into the complete System Test. Performance testing and Regression testing are also key components.

G.1 Unit Test

30. Unit Testing is a procedure used to validate that individual units of source code are working properly. A unit is the smallest testable part of an application. In procedural programming a unit may be a program, function, procedure, web page, menu, etc, while in object oriented programming, the smallest unit is always a

class. The Unit Test provides verification, usually by an independent tester, that the application functionality and features meet the user requirements.

G.2 Systems Test (Integration, Performance, Regression)

31. System testing is comprised of integration, performance, and regression tests. It verifies that no defects are introduced when configurations change or when components are combined to form a release or a new software version. It is conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements – both functional and non-functional. Integration Testing combines the individual components and prior sub-assemblies to validate that the combined components correctly interface and operate on the full hardware system. It falls within the scope of 'black-box' testing and requires no knowledge of the inner design of the code or logic. Performance Testing is performed to determine the speed at which some aspect of a system performs under a particular workload. It can also serve to validate and verify other quality attributes of the system, such as scalability and reliability. Regression Testing is a comprehensive re-test of an entire system whenever the system source code is changed. Regression tests are done to determine whether software changes have inadvertently introduced defects into previously operational application functions. A test that is run on the production environment and targets only major functionality can be done as a final check before releasing the system to production. This is referred to as a 'Smoke Test'.

32. A User Acceptance Test is the final evaluation by users to determine whether a business solution is fit for use and acceptable for release to production. It should emulate real-world usage conditions such as users performing normal work operations. The results of these tests give confidence to the client as to how the system will perform in production.

H. Ongoing Activities

33. A critical component in all software applications is the process that manages changes. Changes to software may be necessary due to missed requirements, a poor design, defects in code, or simply a desired enhancement not initially identified. Software Configuration Management elicits a change control plan that identifies and documents change requests so that the respective manages can gauge impact and jointly determine if the change should be made and when it should be introduced into production. It should always be possible to return to an earlier code version if problems with the new software release occur.

34. As a project moves through its various phases, team members may identify areas of improvement and/or areas of success either in the work process or in the work product. At the end of every project, a Lessons Learned document is created to document these components and to identify and recommend new best practices that should be considered for general implementation. All batch systems will have an operations manual created by the team that will describe how to run the batch system in production.

35. Audits will be conducted by an independent staff to ensure that a project team is adhering to the agreed upon project management and software development standards and procedures. This can be very important, especially in the early stages of adoption, when shortcuts may be considered as deadlines approach. Software Quality Assurance (SQA) provides discipline in the development process and thus instills confidence in the quality of the software.

VI. CURRENT STATUS

36. An effective Software Development Process (Attachment A) is now being used for the Economic Census for the first time. We chose to pilot this SDP effort on all macro processing and dissemination software projects for the 2007 Economic Census. The macro phase of the Economic Census includes all systems necessary to tabulate the data, perform macro level data analysis, apply disclosure avoidance techniques, and prepare the data for dissemination. The preceding micro processing, dealing with data cleansing operations at the individual establishment level, implemented the scaled-down minimum process. It used only essential process components because there was less time available to initiate a major change in the micro work procedures.

37. Although the 2007 Economic Census has only reached the half-way point, the software development process has already proven to be beneficial. The improved quality introduced with a defined and repeatable process has provided a smoother and more timely implementation of the software systems into production. All micro systems were in production on February 4, 2008, the earliest production date ever achieved. Systems functioned very smoothly throughout, especially during the critical processing periods, without any issues. All macro systems were on the shelf several weeks before the 31 October 2008 production date and are currently functioning in production as required, within budget and meeting planned system scope.

VII. LESSONS LEARNED

38. As we have implemented our plans to improve the overall quality of the economic census, we have learned that the quality of software is a key component in the success of all statistical systems. A Software Development Process provides a model for quality improvement, especially for systems with long life cycles. But to be successful, it must be viewed as a continuous process that can adapt and change over time. There is no perfect process for developing software. Given variations in the size of a project, the skills of the team members, the technology used, and the risks that could occur if the software fails in use, no one process can work under all circumstances. In addition, trying to change work habits is a difficult task in itself. Plan for improvements to be incremental, and offer training.

39. Other lessons related to our SDP efforts:

a. Clear, specific standards are essential and should be developed early on in the project lifecycle by a cross functional team. This will ensure that projects start off on the right path from the beginning and that the standards encompass the needs of everyone involved.

b. An independent review and reporting of adherence to standards is valuable but checking for document or process presence is not enough. Document and process quality should also be reviewed to ensure that when project teams are using the standards, they are doing it effectively and as intended.

c. Work smarter, not harder. As long as people keep working in the same way, no lasting improvement is possible. However, change the way people work, their process, and a meaningful and stable increase in productivity can be achieved. Asking people to work harder, that is for longer hours, can provide a short-term boost but in the long term the workers will wear out and productivity will be even lower. Software development is an intellectual activity that requires concentration and diligence. Try to pressure people to finish more quickly and they start taking short cuts, which in the long run makes the project take even longer. Devoting more time to the planning and quality of design and code results in less time required for testing and revision, and instills more confidence and pride in the software system.

d. Allow for errors, but try to uncover them as far upstream as possible. If you demand that people not make any mistakes in their work, then they won't report any to you. Problems will come as a big surprise to everyone at the time of implementation, and the project will be late. But, if you plan for defects in all phases of the process, then you can solve them more easily and efficiently. If you conduct a walkthrough and inspection right after each step of the process (i.e., requirements, design, code) then any defects found come from a more limited set of sources. This allows them to be more easily and quickly corrected than if not detected until the test phase or after the system was in production. Although many may likely rely on testing to improve quality, it is a very costly and inefficient way of removing defects. The more checks that exist for finding defects throughout the process, the fewer will go undetected.

e. Code reviews have offered the best method for cross training and the sharing of programming techniques. This improved collaboration has enabled the entire programming area to improve the quality of their work, beyond the improvements due to delivering a more uniform product with less defects.

f. Separate the 'What' from the 'How'. The elicitation of requirements should focus on ***what*** needs to be done, and can be derived by asking why it is needed or for what purpose. It should be a view from the outside, describing 'what' to do, instead of from the inside, describing 'how' it works. Thus, the requirements are held separate and independent from the design, although the process will be iterative.

g. Forming a proper team environment for the project is critical. A group of individuals assembled together in a room is not a team. In order for those individuals to jell together into a team, they need to trust each other, interact freely, resolve problems together and have everyone's participation. Team members can draw on each other's strengths, and can help overcome each other's weaknesses. People want to work on a jelled team. They have a sense of belonging and a purpose. Even if the project is not very inspiring, they can still inspire each other to new heights of accomplishment. The SDP has worked to create jelled teams by holding what is called a team launch. When a significant project gets underway, everyone on the team works together in the same room for several days to create a plan from which the team will work. Since the team creates the plan together, they all have joint ownership and are more likely to commit to making it successful. Team members are assigned roles which encourages everyone to participate. After the launch, the team is kept together by having weekly meetings to plan work on a continuing basis and to raise and solve any issues, and will conduct re-launches every 3 months or so to scope out the next phase of the project.

h. Keep it short. People can only keep so many pieces of information in their head at one time. Use templates and examples to convey the basic goals of each process. Advocate having short standards that are actually followed compared to long standards that cannot even be remembered.

VIII. NEXT STEPS

40. The success of this project laid the foundation for further expansion of this software development approach throughout the entire Economic Directorate of the Bureau of the Census. It is called the Economic Software Process Improvement (ESPI) program. The process elements are very similar to those of the SDP, and are depicted in a 'swim-lane' flow diagram in Attachment B. This diagram more accurately represents the interrelationships between all of the process elements.

41. Although there is still a significant amount of work left to be done on the 2007 Economic Census, some of the next steps are already apparent. Our goal is to prepare for the 2012 Economic Census while more fully implementing the ESPI process. Because of its similarities to the SDP, the changes needed to comply with ESPI are expected to be minimal. Even so, efforts to continually improve our process will necessitate that we delve further into requirements elicitation and system design than we were able to do for the 2007 Economic Census.

42. For all macro projects, the final rounds of software quality audits will be complete by April 2009. Lessons learned sessions will be conducted and improvements will be recommended for the software development methodology that will be used for the 2012 Economic Census.

IX. CONCLUSION

43.We have not completed all activities for the 2007 Economic Census, but it is clear that instituting a software development and management process has proven valuable to all stakeholders of the program. A defined and repeatable work process has improved quality, communication, and documentation while minimizing the amount of time and resources usually spent on re-work. As a result, the 2007 Economic Census continues to operate under budget while meeting or exceeding all deadlines. Building upon this success, the Economic Software Process Improvement (ESPI) approach will be developed and implemented corporately across the entire Economic Directorate. ESPI will generally cover the same processes depicted in this paper and used for the Economic Census, but will provide even more direction, templates, examples, and standards. ESPI will be

more comprehensive, but is also planned to have the flexibility to support the various Directorate programs. We anticipate ESPI will revolutionize the way software is developed in the Economic Directorate. The goal of ESPI is to deliver software code into production that works correctly and meets users' needs…Every Time. We believe we are on a path that will improve the efficiency of our operations and ultimately the quality of economic statistics.

References

Hiller, David (2005), "Post Collection Software Development Process", unpublished report, Washington, D.C.,USA: United States Census Bureau.

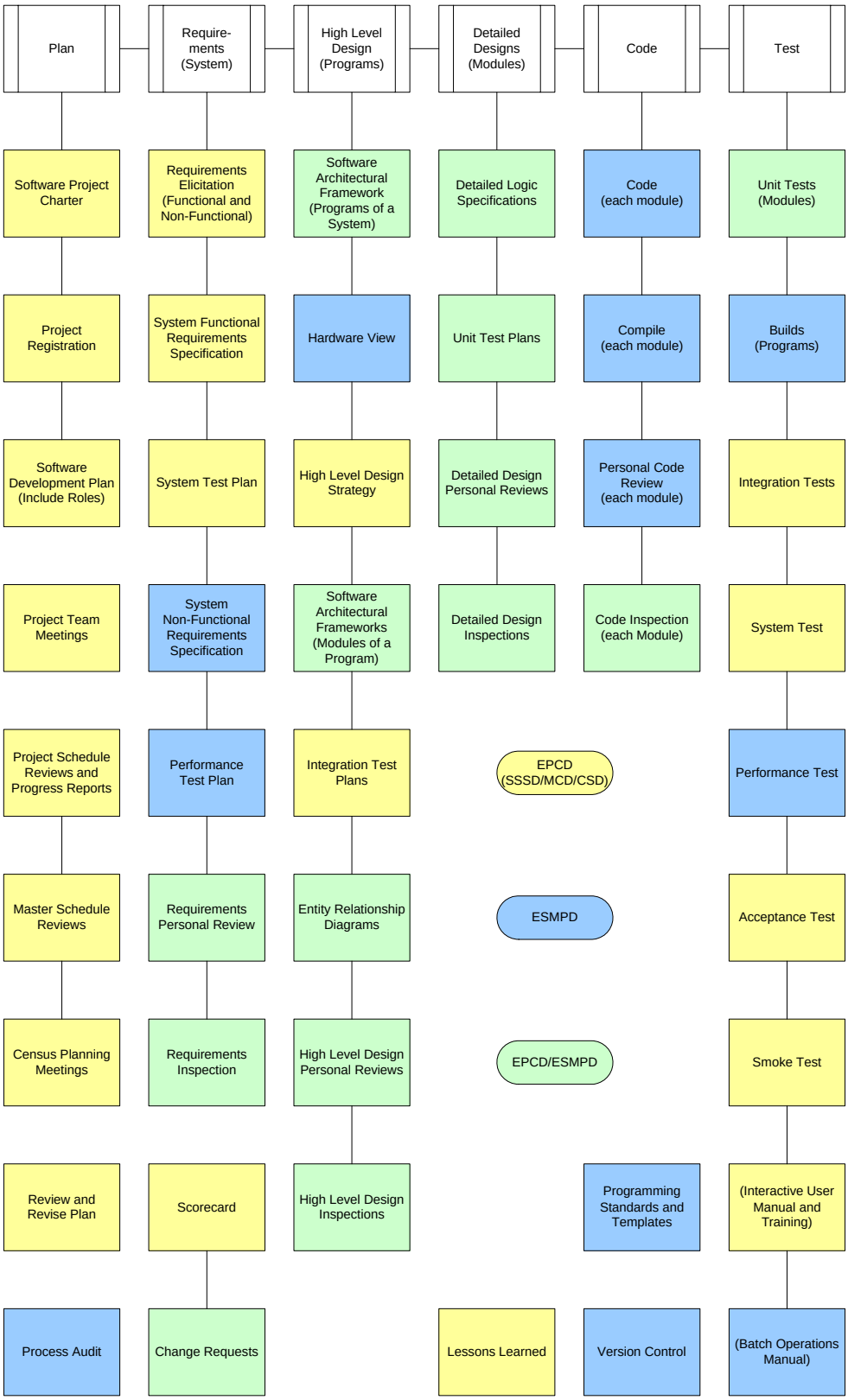
Stempowski, Deborah and Chapman, David (2007), "Using Quality Audits to Improve the 2007 Economic Census", The European Conference on Quality in Official Statistics, Washington, D.C., USA: United States Census Bureau.

Walker, E. (2005), "2007 Economic Census Program Plan", unpublished report, Washington, D.C., USA: United States Census Bureau.

Attachment A

Recommended Process

David H. Hiller 4/17/2006



Attachment B