Whitemarsh
Information Systems Corporation

Business Information System
Life Cycle Costs

Whitemarsh Information Systems Corporation
2008 Althea Lane
Bowie, Maryland 20716
Tele: 301-249-1142
Email: Whitemarsh@wiscorp.com
Web: www.wiscorp.com

Table of Contents

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

1.0 Objective

The objective of this short paper is to show how the traditional system development life cycle can be enhanced with prototyping and business information system generators to increase productivity and quality while at the same time reduce cost and risk. To accomplish that end, this paper addresses the:

- Traditional System Development Life Cycle
- Enhanced Traditional System Development Life Cycle with Prototyping
- Employment of Function Point analysis to create accurate and valid project estimates
- Effect of business information system generators on prototyping, and virtually all other phases of the System Development Life Cycle.

2.0 Traditional System Development Life Cycle

Figure 1 illustrates the main phases associated with the traditional first implementation life cycle of an enterprise-wide business information system such as human resources, or accounting and finance. The scale on the bottom represents the percent of completion across all phases. The purpose of the chart is to show the relative quantities of staff hours expended across time.

In general, if the costs associated with requirements and design are $1, the activities associated with detailed design through initial business information system implementation costs another $4. That's $5 in total for a first implementation cycle. Figure 1 shows that:

1. About 70% of all effort is expended prior to the first real demonstration of a business information system. That is, before the Implementation, Operation and Maintenance parts of the last major phase. This percent may be much higher if requirements-changes are discovered during System Test. Some multi-hundred million dollar efforts are scrapped because requirements change too dramatically before System Test.

2. The 2nd and any subsequent business information system versions require recycling of the first two phases (20-60%). That is, a repeat of part or all of Requirements Analysis and Design, and also Detailed Design, Coding, and Unit Testing.

In addition to the two take-aways from Figure 1, the very problem for which the business information system may have been developed may have changed too much the first production implementation is complete. This "hazzard" is what the United States Government

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

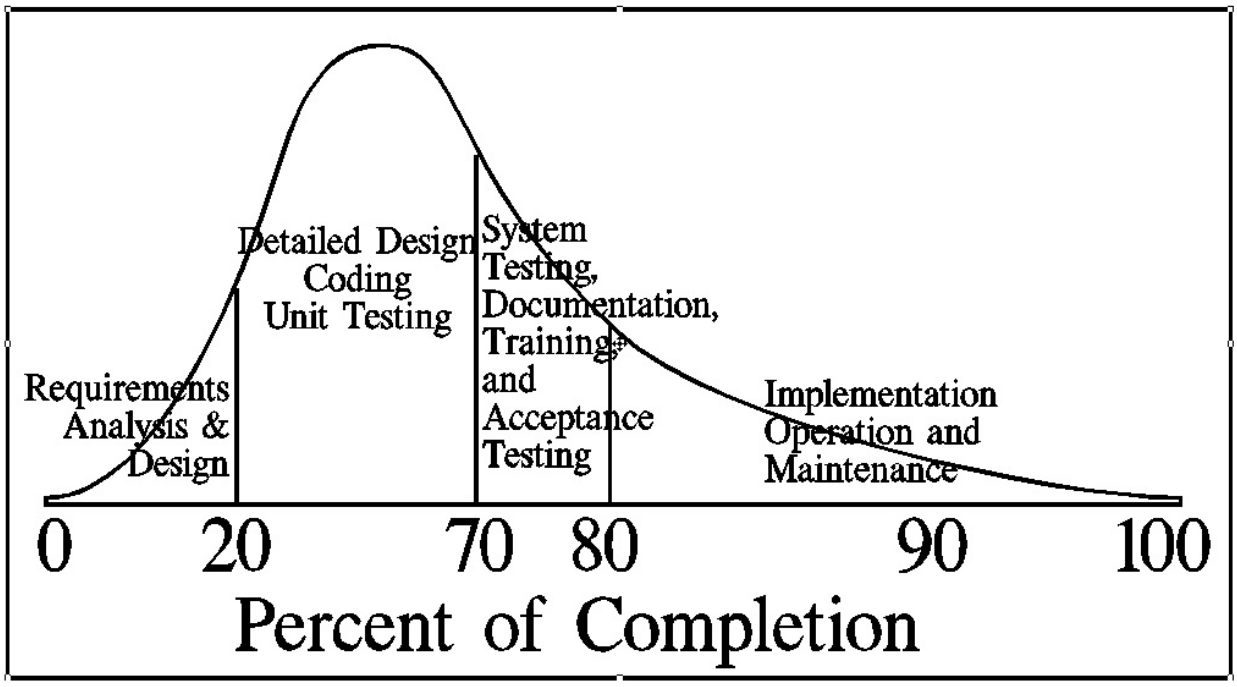

Figure 1. Traditional System Development Life Cycle (SDLC).

Accountability Office concluded is intrinsic to the very process of business information system development. If not addressed, this can result in perpetual recycling of requirements without ever getting to System Testing.

Beyond the first-cycle implementation cost, the total life cycle expenditure for business information system revision cycles (not shown in Figure 1) commonly costs five times more. The total life cycle cost is thus, 30 times the design cost.

The problem is not that requirements change, however. Rather, the real problem is that the effects of requirement changes that occur once System Testing is complete are too costly to be reflected in the first version of the implemented business information system. Hence they are left to a "maintenance cycle."

It's a given from the very start that requirements will change, even though this assumption is seldom folded into methodologies. Requirements-change can be especially fatal to procured software packages if these packages have not been designed for change from the very beginning. The holy grail to be achieved therefore is to largely if not completely eliminate the need for changing requirements prior to the very first activity of production business information system development.

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

3.0 Enhanced Traditional System Development Life Cycle with Prototyping

Because $1 in requirements' changes can cause up to $29 in additional life cycle costs, the exhortation is simple: Get the requirements right the first time because the cost of change is prohibitive. The reason the costs are so high and the time is so long is that:

- The tools that either dramatically shorten, or even eliminate steps have only recently become demanded and/or developed

- Traditional data modeling is employed in isolation rather than coupling it with prototyping.

- Not recognizing that if iterative development is contemplated, whenever required data model changes are needed to accommodate a subsequent business information system iteration may, in turn, require production code changes in prior business information system iterations that are already in production.

If a data-driven methodology is adopted and is supported by a quality business information system generator, the steps, shown in Figure 1, "Detail Design...," and "System Testing..." can be largely eliminated. If a quality metadata management system (e.g., the Whitemarsh Metabase System) environment is installed that stores and manages all the SDLC work products in a highly engineered, reusable form on an enterprise-wide basis, there can be a dramatic improvement in the productivity and quality of the first phase, Requirements Analysis and Design.

Finally, if we implement after four to six design cycles of a prototype, whole cycles of maintenance can be eliminated because several such cycles are really to implement the recently uncovered requirements rather than make needed changes..

Quality business information system generators such as Clarion (www.SoftVelocity.com) can import a database's design and generate a working business information system in a few hours. From that first generation, the generated business information system can be "electronically pruned" to a form that is suitable for demonstration. After several iterations of demonstration, modification, and regeneration, a quality specification can be created.

The critical change to the traditional process is illustrated in Figure 2. Shown are four prototype development cycles, and one full-implementation cycle. The first "P" (i.e., prototype) cycle contains the activities noted (that is, Design, ... , Feedback).

Each subsequent "P" cycle is dramatically reduced because it is focused only on design changes, regeneration, and the next demonstration. The first cycle contains a requirements analysis and design step. But, that step is reduced to mainly producing the data model.

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

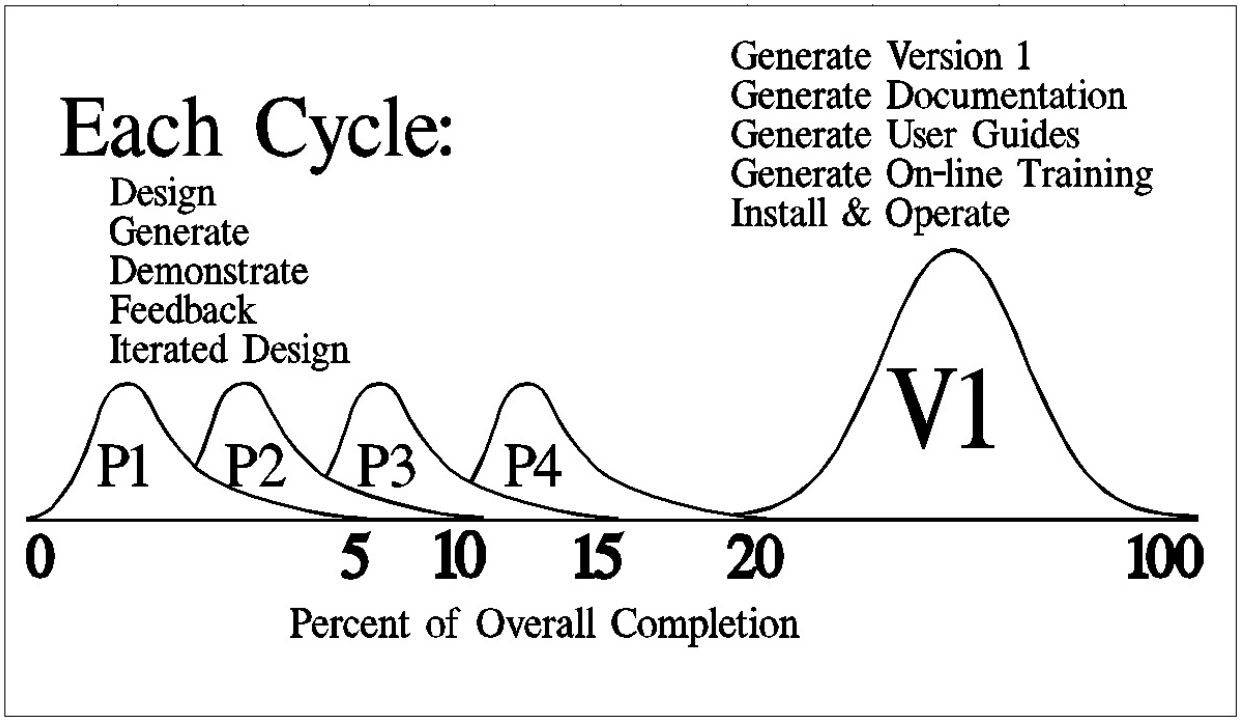

Figure 2. Traditional System Development Life Cycle with Prototyping.

To arrive at the prototype, the business information system generator imports the data model and generates the first-cut business information system. A day or so is spent to make it "pretty." Then, the prototype is demonstrated. In all, only several weeks will have passed. Each "P" thereafter can occur in one or two weeks. Under this changed approach:

- "Real" business information systems are available to run at the end of every "P" cycle.
- Requirements are iterated until fully known, but are determined in a very condensed time-period.
- Full development occurs only after requirements are completely validated.

Under either case, the total life cycle cost is the cost of the first cycle plus five times more to account for revisions and extensions. How these affect the maintenance cycles is addressed in Section 5.

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

The costing is shown in Figure 3. Under the traditional approach, if the first box costs $400,000[1], the total life cycle costs are 30x, that is, 5 + (5 * 5) or $12 million.

Faced with high costs, the response to the common the high-pressure demand for immediate and visible results, is to trim the first box, that is, requirements analysis and design. Suppose it was trimmed by one-half. In theory, by trimming the first bos, the total cost of the first business information system implementation cycle would be reduced to 2.5, that is, 0.5 + (4x 0.5). The total life cycle cost would only be reduced to 15 units, or $6 million. That's a 50% savings. Such savings are however false.

That's because the real requirements are still undiscovered. The only thing that has changed is that the analysis to discover the real requirements has been reduced by 50%. In short, 50% of the real requirements are undiscovered.

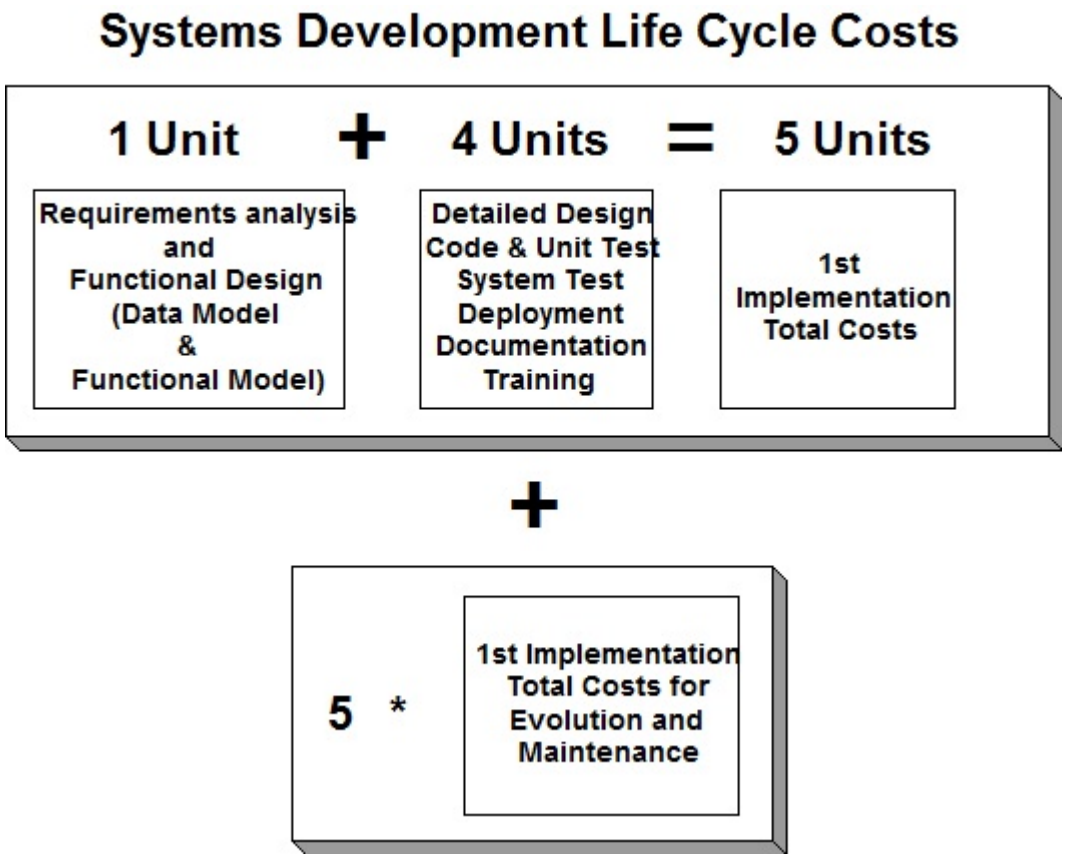

Figure 3. Cost structure of Systems Development Life Cycle.

[1] This represents the cost of a 250 table database-based business information system that is computed via Function Point Analysis set out in Section 4.

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

If the requirements and design efforts are reduced by 50% from 1.0 to 0.5, and given the United States Government Accountability Office (GAO) error allocation of 41% of all information technology errors occur if this step is underdone or done wrong, the "4x" (the amount to accomplish what is really "required") is likely to be 12x.

That is because the business information system would have been done incorrectly. Consequently, the incorrect portions of the business information system have to undone and then done right. Hence, three times the "4x" number. In that case, the 1st business information system implementation cycle cost is not the original 5 nor the 2.5, but is 6.5 (2.5 (done wrong) + 4.0 (now done right)).

The overall life cycle unit quantity becomes 39, that is, 6.5 + (6.5 * 5). This is really a "pay me now, or pay me more later" situation. If a unit costs $400,00, the life cycle cost is now $15.6 million instead of the original $12 million. In short, a requirements analysis of design cut of $200K results in an overall cost increase of $3.6 million. Clearly, these savings are costly indeed.

For a real impact, the savings must be made to the 4x component (that is, Detailed Design,... , Training). If the 4x component is reduced to 2x, the life cycle costs are reduced to $7.2 million, that is, 400K * 3 + (400K * 3) * 5.

In addition, if due to the prototyping approach, three of the five major business information systems evolution cycles are eliminated, the whole life cycle cost is reduced to $3.6 million, that is, 400K * 3 + (400K * 3) * 2. That represents dramatic savings of about 66% from the original $12 million. Now, in contrast to the $200K savings that cost $3.6 million, these are real savings.

Simply, this means that three business information systems can be implemented for the price of one, or that productivity goes up by 300%. While this is a compelling case, can it be done? Yes, and this has been standard best practice for the data-driven Clarion community for the past 20 years. The steps for a quality business information system generator approach are these:

- Import the database design.
- Generate a first-cut application.
- Electronically prune the first-cut application metadata-based design to the desired behavior model.
- Generate the prototype for end-user evaluation.

Thereafter, present the prototype to users, gather changes, and repeat the steps above until the users say, "enough already." Then, and only then, create the production version. Once the production version is created, there are the evolution and maintenance cycles.

4.0 Employment of Function Point analysis to create accurate and valid project estimates

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

Function point analysis is a technique for estimating the resources required for the development of a business information system. It accomplishes the development of the estimate by:

- Determining the types of business information system components that must be constructed.
- Estimating the quantity of each component to be built
- Assigning a function-point count to each component for simple and complex versions.
- Computing the total quantity of function points
- Multiplying the function point count by a value assigned to each different software development environment that reflects its cost to construct a function point.

For business information systems, the types of components that are to be constructed are:

- Third Normal Form Data Model Tables
- Screens
- Browses within Screens
- Menus
- Third Normal Form Processes that are hand coded

For each, an example of one that is simple or complex is set out. The function point count for each is assigned. For example, 4 function points for a simple component and 8 function points for a complex component.

At that point, experimentation is conducted to determine, for a given development environment, how many staff hours needed to create a typical component.

While this appears simple and straight forward, the process presumes that the function point analyst understands the business information system requirement in such detail that the quantity of each of the five components can be quantified to a high degree of accuracy. Frankly, that is close to impossible. So the question becomes how to determine the quantity of function points? Clearly, what is needed is a quick step and then a short-cut.

Over a period of years, Whitemarsh has been creating function point totals by counting the actual quantity of produced components AFTER the business information system is complete. Whitemarsh divided the determined quantity by the count of third normal form database tables. The resulting number was around 80. The short cut is this: create a third normal form data model for the business information system and multiply the count by 80. The resulting quantity is the total number of function points, +/- 10%.

That, however does not produce cost. Whitemarsh has also been keeping accurate records of the quantity of hours it needs to build business information systems using Clarion. At the rate

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

of $100 per hour, the computed cost of a function point for Clarion is $50. This compares quite favorably with traditional business information system development environments of about $250 per function point.

5.0 Effect of business information system generators on prototyping, and the SDLC

Over time the first production implementation, no matter how expertly or efficiently implemented must undergo maintenance. With this approach, maintenance cycles should be both fewer and less costly. That is because the approach has positive impacts on the three most significant maintenance cycle cost contributors, which are:

- Degradation of the initial implementation architecture, engineering and coding due to ad hoc changes and patches.
- Staff turnovers that often result in the "NIH" syndrom
- New features and functions

First, as systems developed through traditional design and coding techniques get "old," their designs and software constructions degrade through the application of ad hoc patches and quick fixes. Because it takes much longer to accomplish maintenance work than new construction work. The result is the same volume of work because while there are fewer changes, each takes longer.

In general there are two types of changes: Database design, and custom coding. If there is a high quality database design, it is often independent of organizational peculiarities. That is because it has been designed to represent the data and interrelationships required to meet enterprise mission requirements. Business information systems constructed from a database engineering foundation requires fewer changes over time. This almost automatically ensures great stability in the fundamental architecture of the database's application software.

For coding changes, upwards to 95% of all business information system code can be generated. Thus, what needs to be changed is either the code that is generated or the code that is custom developed. If the former, the cost is close to nil. If the later, the cost is greatly reduced because the quantity of code lines is reduced by 95%.

Second, staff turnover. New staff have to learn the system from scratch. That requires an almost complete recycle of the Figure 1 processes, which is both painful and long. In addition, when there is the not-invented-here (NIH) syndrom to contend with, every maintenance task gets transformed into a fix and enhance task. This naturally takes longer and the enhancement often installs new bugs.

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

While this problem is, of course, unavoidable, if upwards to 95% of the business information system's computer program code is generated, it does not have to be learned. In fact, because the ultimate program's code is all but invisible, it never has to be addressed at all.

Quality business information system generators such as Clarion operate at a meta-design level. Computer program software code is generated from its meta-design level. Changes are made at the meta-design level as well. So, once changes are made, the ultimate computer program code is regenerated.

Third, as new features are required, some may be design breakers. Because a production system will have already been implemented, changes to the existing system will take longer because there will be redesign, recoding, and the tedious and error prone data conversion.

This third type of evolution and maintenance problem largely disappears because there are four or more prototype cycles before first implementation. Thus, maintenance caused by an immature design are eliminated because the design has matured through prototyping before it is implemented.

Because the three maintenance cycle problems cited above are addressed, the generally occurring five maintenance cycles are reduced to three and becomes quicker to accomplish.

Figure 4 presents the overall summary of the approach described in this paper. The left side of the figure shows that 33% of the overall effort is allocated to the creation of the Version 4 design.

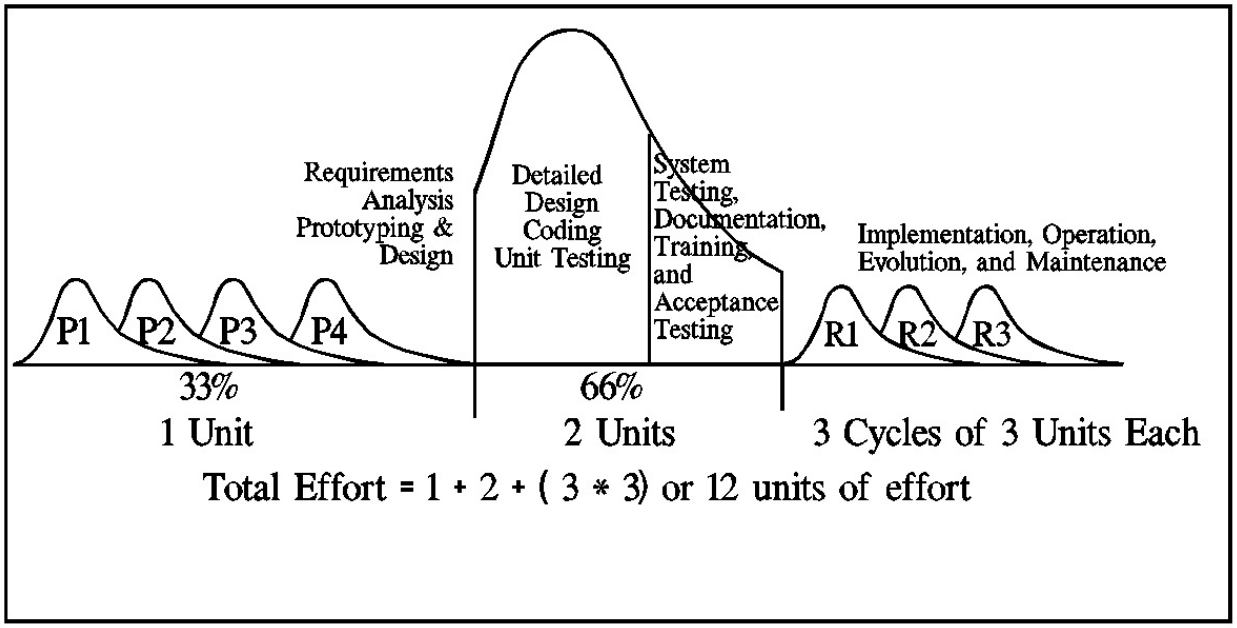

Figure 4. Overall effect of applying the Whitemarsh approach to Business Information System accomplishment.

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved

A Version 5 design results from the feedback on changes that occur after prototype #4. 66% represents the amount of effort that must be expended to accomplish implementation of the first production version. Subsequent to that, there are three cycles of maintenance instead of the usual five. Each is really a single "P" and a production implementation cycle using the code generator. The result is three cycles of 3 units each. The overall unit quantity is thus 12.for the complete life cycle of the specific business information system.

6.0 References

The following references to Whitemarsh materials provide a more detailed exposition practical application of the significant content of this paper. The following documents are available free from the Whitemarsh website:

Paper	URL
Enterprise Architectures	www.wiscorp.com/sp/sp08.pdf
Engineering and Managing Information Systems Plans	www.wiscorp.com/sp/sp11.pdf
Data Modeler Architecture and Concept Of Operations Reverse and Forward Engineering Guide Metabase Module User Guides	http://www.wiscorp.com/metabase_demo.html
Metabase Overview	http://www.wiscorp.com/metabase.zip
Modeling Data and Designing Databases	www.wiscorp.com/sp/sp06.pdf
Function Points Strategy for Business Information Systems Estimating	www.wiscorp.com/sp/sp13.pdf
Whitemarsh Approach Consequences	www.wiscorp.com/sp/sp27
Challenge for Business Information System Success	www.wiscorp.com/sp/sp19.pdf
Data Models: The Center of the Business Information System Universe	www.wiscorp.com/sp/sp20.pdf
Metadata The Information Technology Intellectual Property Of The Enterprise	www.wiscorp.com/sp/sp21.pdf
Database Objects Introduction	www.wiscorp.com/sp/sp22.pdf

Copyright 2015, Whitemarsh Information Systems Corporation
Proprietary Data, All Rights Reserved