A temporal information system cost model in supply chain management

Dai, Yuyu

A Temporal Information System Cost Model in Supply Chain Management

by Yuyu Dai

12th November 2009

Summary

In the supply chain realm, an information system plays a crucial role in defining its capacity and efficiency. As the supply chain itself evolves constantly due to strategical or tactical business optimisations from supply chain participants, information system planners experience tremendous challenges in defining cost-effective and flexible supply chain information systems to align with existing and forthcoming supply chain changes. The principal challenging question is how much the supply chain management system will cost against the functions delivered.

In many cases, computer system planners rely on software suppliers' quotations solely to estimate the supply chain system cost. However, a quotation hardly suggests the precise total cost due to lack of understanding or information of the particular implementation. Apart from this reason, there are usually various hidden costs that are relevant to the users only, such as unplanned changes. These costs will certainly not be reflected on the quotations. A practical supply chain management system cost estimation method is therefore indispensable to the system planners.

This research is an attempt towards estimating the supply chain information system cost at the early stage by synthesising resource constraints of budget, time, implementation scheme and developing team. A temporal cost estimation model is proposed to facilitate effective operational decision making on supply chain management system practices, in regard to the high supply chain uncertainty. With the mathematical cost model in place, analytical researches are performed to complete the knowledge base that a supply chain management system planner would need. Finally, the model strengths and weaknesses are reviewed for reference and future improvements.

Acknowledgments

This thesis is prepared for the Master's degree by research as part of my Knowledge Transfer Partnerships (KTP) project. KTP is a government funded programme to improve the productivity and competitiveness of business organisations through the better use of knowledge, technologies and skills that reside in the academic institutes. My KTP project was established by the University of Durham in conjunction with the CAV Aerospace Limited to construct an electronic data exchange software system, in order to help the aerospace manufacturer to facilitate effective communication with other partners in the global aerospace supply chain. As the KTP associate, I benefited from the project work as well as the KTP's excellent personal development programme in areas of career development, management skills, networking with other associates, etc. In terms of academic research, the provision of a KTP project management committee also made my degree study better than ordinary postgraduate studies due to the abundant source of experience and knowledge that I could rely on.

I would like to express my gratitude to the KTP organisation, University of Durham and CAV Aerospace Limited for giving me permission to commence this study in the first instance, to do the necessary research work and to use various resources. During the course of my study and KTP project, I received vast supports from the project committee, which consisted of members from academia (Dr Hui Long, Dr Oliver Vogt, Dr Dirk Schaefer, Dr Peter Matthews), industrial partners (Mr John Lyle, Mr Owen Mcfarlen, Mr Peter Carssidy) and KTP representatives (Mrs Val Wooff, Mr Richard Parker Smith). Also, I would like to give my special thanks to all others from the University of Durham and the project host company who provided convenience to help to progress my project over the last two years.

I am deeply indebted to my supervisor Dr Hui Long from the School of Engineering at Durham University, whose incentive advices and encouragement helped me to overcome many difficulties throughout the study. Dr Hui Long took over this supervisor's role after Dr Dirk Shaefer's departure. Despite the difficulty that my thesis topic was outside her research scope, she spent great effort in supporting me from various areas. She helped me to participate the HAAMAHA 2007 international conference so that I could present my research to the public. Without her endeavors, I would have never been able to complete the study.

Furthermore, I have to thank my industrial supervisor, Mr John Lyle, who initiated lots of background work in advance to facilitate my commencing and progressing the project without much difficulty. John not only fulfilled his designated role, but also provided substantial supports to my study.

Extra heartfelt appreciation should also attribute to Dr Oliver Vogt, without whom the project might have never been precipitated. For a period, I was hesitating in deciding the direction of my research. Oliver spent a great deal of time to enlighten me for possible research areas and research methods. His effective liaison among KTP partners eliminated many potential problems so that I was able to focus on my project and study.

Specially, I have to give my sincere acknowledgment to assistant professor Judith Gebauer and her research team in the University of Illinois at Urbana-Champaign who motivated my research with their innovative publications. Their informative feedback to my contact intensified my understanding of this research area therefore sped up the research progress.

Finally, I would like to express my sincere gratitude to my wife Susie whose patient love enabled me to complete this study. We live thousands of miles away from our home, her support and encouragement was the primary impetus that moved me forward every day.

Contents

List of Tables

List of Figures

List of Acronyms

ANOVA Analysis of Variance Table

BAM Business Activity Monitoring

COCOMO COnstructive COst MOdel

COTS Commercial-Off-The-Shelf

CRM Customer Relationship Management

DOE Design Of Experiments

EAI Enterprise Application Intergration

EDI Electronic Data Interchange

ERP Enterprise Resource Planning

FTC Flexibility To Changes

FTU Flexibility To Use

ISCM Internal Supply Chain Management

KPIs Key Performance Indicators

MIS Management Information System

OAT One factor At a Time

OSS Open Source Software

RAD Rapid Application Development

RFID Radio-Frequency IDentification

SA Sensitivity Analysis

SC Supply Chain

SCM Supply Chain Management

SCMS Supply Chain Management System

SRM Supplier Relationship Management

TMF Transaction Management Foundation

VBSE Value-Based Software Engineering

XML eXtensible Markup Language

Chapter 1

Introduction

In this chapter some background of the supply chain and its correlation with information systems is explained. A structured approach is followed to address the problems and to give an overview of this study.

1.1 Supply Chain and Supply Chain Management

Supply Chain (SC), as the successor of the logistic network towards the global trade popularisation, attracted public and research attentions in the late 1980s and later came into pervasive practices in the following decade (Oliver and Webber, 1992). Although the term "supply chain" is publicly recognised, its formal and unified definition still has not been reached. Up to now, several recommendations have been proposed from slightly different perspectives such as (Quinn, 1997; Lambert et al., 1998), and also:

> *A supply chain consists of all parties involved, directly or indirectly, in fulfilling a customer's requests. The supply chain not only includes the manufacturers and suppliers, but also transporters, warehouses, retailers and customers themselves.*
>
> (Chopra and Meindl, 2004)

The supply chain has been recognised as an influential economical evolution enabled by information technologies. It encourages partners to coordinate closely through in-depth information sharing to facilitate agile interactions and minimise transaction overheads (Premkumar, 2000).

In general, each supply chain participator falls into certain supply chain stages including planning, sourcing, manufacturing and delivering, but most of them share the same key point in approaching effective supply chain practice: managing relationships and coordination in order to survive and thrive in the networked economy. The activities to achieve this target are referred to as Supply Chain Management (SCM). Once again the SCM can be defined from material, product, management philosophy or management process point of view. The Global Supply Chain Forum[1] proposed a definition of SCM in year 1998 as:

> *Supply Chain Management is the integration of key business processes from end user through products, services, and information that add value for customers and other stakeholders.*
>
> (Lambert et al., 1998, p.504)

[1] A research group in the Ohio State University reformed in January 1999, directed by Dr. D.M. Lambert.

From the business management perspective, SCM covers activities and coordination of the whole production life cycle in an integrated network that comprises of manufacturers, suppliers, distributors, customers and all other organizations involved (Hugos, 2003).

1.2 Supply Chain Management System

Despite the fact that each market has distinct characteristics that a supply chain serves, there are many demands in common which define the capabilities of a supply chain. Chopra and Meindl (2004) identified five major supply chain performance drivers (areas that supply chain management should focus on) as shown in Figure 1.1. The four direct drivers (i.e., Production, Inventory, Transportation and Location) are inter-connected by an information system to form a star topology. Each connection (interface), represented as a solid arrow on the diagram, is comprised of a set of business processes. It is also noticeable from the diagram that the conventional business processes (dotted line arrows) among the direct drivers are replaced by connections with the central information driver to improve the efficiency, particularly for large and complex supply chain. This indicates that information system is playing a pivotal role in a supply chain network.

At the early stage, a Supply Chain Management Systems (SCMS) were regarded mainly as a decision support or analytical tool to guide the other four direct drivers. Along with the progress in information technologies and SCM research itself, information system has extended the original role to become an all-around supply chain enabler.

According to the influential book of *Supply Chain Management : Strategy,*

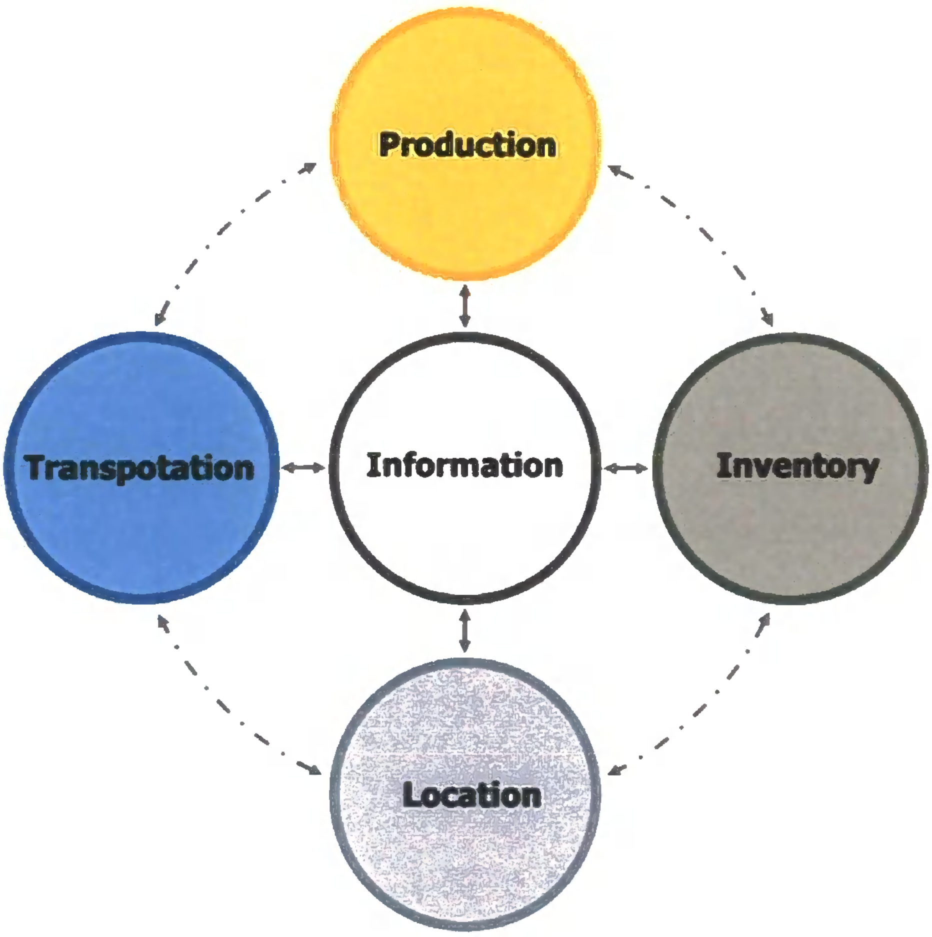


Figure 1.1: Five major supply chain drivers (Chopra and Meindl, 2004)

Planning and Operation (Chopra and Meindl, 2004), supply chain management practices should not only improve internal performance, but also intensify focus on the total profitability of the supply chain as a whole (i.e., to value other partners within the supply chain by cooperating seamlessly), which include four macro processes:

- Customer Relationship Management (CRM)
- Supplier Relationship Management (SRM)
- Internal Supply Chain Management (ISCM)

- Transaction Management Foundation (TMF)

Due to the rapid progress of information technologies and their application at the above four process levels, SC has been more and more referred to as "e-SupplyChain". This movement resulted in prosperous stories in various areas but also led to the dilemma of handling high uncertainty and constant changes effectively since most management information systems were designed to handle determinate business processes only. Changes from the processes will result in the adjustment of information system inevitably.

In most cases, organizations tend to apply Commercial-Off-The-Shelf (COTS) software products or buy turnkey systems from software vendors due to the considerable complexity of software development and the sophistication required that cannot be delivered by themselves. However, owing to the fact that each supply chain organization is "unique" (different from other organisations) and "dynamic" (changes over the time) in terms of operational details from the software perspective, it is likely that ready-made software products cannot satisfy sufficient current and future demands. This is especially the case in the supply chain context, where functional gaps always exist when integrating with new partners.

A supply chain management system is expected to align with business processes all the time to constitute a consolidated Supply Chain ecosystem, where changes from one point propagate to other nodes across various supply chain organisations. The affected nodes including their associated SCMSs need to accommodate the influence of changes so that the supply chain can stabilise again. This indicates that supply chain changes come from both internal and external sources (the networked supply chain) therefore its software system uncertainty is much higher than a normal information system. For this reason SCMS needs to be flexible to keep the cost of system updates low.

Due to these SCMS characteristics, traditional software engineering techniques are not suitable to undertake the SCMS cost estimation. It is highly demanded to have a practical cost estimation method to direct the planning and implementation of a SCMS.

1.3 Research Objectives

It is not the intention of this research to originate a brand new information technique to cope with the supply chain uncertainty: it might be possible in the future but the current software industry aims to deliver to definitive demands. It could still be decades away from bridging this gap. The purpose of this research is to answer the straight-forward yet difficult questions of: What kind of information system should we have to support a supply chain under the conditions of high uncertainty and variability, and how much will it cost? These questions will be answered with a quantitative approach by developing a theoretical cost estimation model, followed by system implementation studies based on the model to assist the decision making. This cost model is inspired by the process model of Gebauer and Schober (2006). It anatomises origins and influences of information system support by inspecting the cost transition of a generic single supply chain process as well as multi-process cost composition. Statistic abstractions are followed to achieve a practical cost model.

In Chapter 2, a literature review is conducted to establish the overview of the state-of-the-art research initiatives and industrial practices in this area, including a description of the process model proposed by Gebauer and Schober

(2006), which motivates this study in many ways. Chapter 3 presents the detailed temporal cost model by progressive logical and mathematical reasoning. In Chapter 4, standard sensitivity analyses are performed to demonstrate influences of various cost model variables to the final cost, by means of the acquisition from Chapter 3 - the temporal cost model. In addition to the model based analysis, Chapter 5 traverses significant SCMS implementation topics by case studies and analysis from system planners' perspective, to develop practical experience in achieving cost effectiveness by means of the cost model. The strengths and weaknesses of this research are summarised in Chapter 6 to complete the overview and to suggest possible future work. The last chapter reviews the research deliverables and findings that have been achieved.

Chapter 2

Literature Review

This chapter reviews state-of-the-art research initiatives and current industrial practices in the relevant area, to seek the possibility of applying them to achieve the objectives of supply chain cost estimation. In particular, Gebauer and Schober's cost model is analysed in detail for the strengths and weaknesses, which compose the background of the new cost model in the following chapter.

2.1 Supply Chain Uncertainty

In the early 20th century, The Ford Motor Company managed much of what was needed to feed its vehicle manufacture lines - from ore mines to timber yards and even to farms in order to produce the raw materials of car interior fabric on its own (Ford, 1926). In this way, this manufacturer constituted a vertical-integrated mass production conglomerate. This broadly mimicked mass production FORD MODE (Batchelor, 1994) achieved tremendous success at that time due to improved efficiency and minimised uncertainty in the production cycle. However, several decades later, this production mode finally faded out subject to very poor flexibility in providing diverse varieties that the market demanded. Henry Ford summarised this infirmity in his own biography (Ford and Crowther, 2005): "Any customer can have a car painted in any colour that he wants so long as it is black".

The remedy to the Ford Mode was a decentralised production network within which each organisation concentrates on single or limited functions available to others. In this way, organisations were also able to provide flexible products, with reduced lead times, inventory level and costs from transportation and material procurement by balancing a range of suppliers and customers. This structure, as the rudiment of the modern supply chain, biased focus but also magnified the uncertainty simultaneously since organisations were free to choose partners in a dynamic fashion. The freedom of switching requires the ability to process large amount of information and deliver product variables in a short lead time. However, each adjustment to align with the external changes incurs extra cost and time. At the supplier side, demands are also very difficult to predict and forecast accurately due to the phenomenon of information distortion (termed as THE BULLWHIP EFFECT by Lee et al. (1997)), as well as other characteristics specific to its particular industry. In a technical report by Muckstadt (1997), comprehensive examples were studied and concluded to the same results.

Supply chain management was therefore recognised as a research area in management science to balance resources, information, structures, performance, etc. in the supply chain. In particular, the information system is one of the key drivers to achieve simplicity and effectiveness of SCM as discussed throughout this study.

2.2 Supply Chain Management System Flexibility

The substantial complexity, uncertainty and capricious collaboration of a supply chain (Milgate, 2001) impacts the supporting information system directly. The challenges could come from insufficient information system functionality, poor usability results from excessively configurable environment, or lengthy yet expensive system developing cycles. SCMS planners need a method to balance the cost with system capabilities. Applegate et al. (1999) defined the ability of a supply chain management system to accommodate procedural and data changes as *flexibility*, and this ability was usually assessed from the cost and time perspective.

In the well-known supply chain council's SCOR[1] module, supply chain flexibility was defined as:

> *"The agility of a supply chain in responding to marketplace changes to gain or maintain competitive advantage."*

In plain words, flexibility is about adaptability to changes. This assertion was backed up by even earlier research initiatives from different areas such as manufacturing (Upton, 1994) and finance (Trigeorgis, 1993). To take the most advantage out of a supply chain, information systems are expected to be flexible at low cost.

[1] The Supply Chain Operations Reference-model (SCOR) is a process reference model that has been developed and endorsed as the cross-industry standard diagnostic tool for supply-chain management. More information is available on the website http://www.supply-chain.org/

Flexibility has been regarded as an important strategic driver of supply chain development in the future (Lee, 2004; Morgan, 2004). From the management point of view, information system coverage, cost and life time need to be measurable in the first place as references to the overall supply chain implementation. In reality, information systems can also contribute to customer satisfaction, output, efficiency, and shareholder value (Hammer and Champy, 1993). Taking the supply chain life-cycle as a whole, most of these characteristics are interconnected with cost under the premise of delivering designated performance (Kauffman and Walden, 2001; Gunasekaran and Ngai, 2004). Indeed, it is the balance of costs and benefits that drives the application of information systems in the commercial realm.

The term *flexibility*, according to Hanseth et al. (1996), denotes either (i) flexibility to further changes or (ii) flexibility in the pattern of use. They are termed as Flexibility To Change (FTC) and Flexibility To Use (FTU) respectively. In the SCMS context, a FTU process can be implemented based on existing SCMS infrastructure without affecting other processes. A FTC process type, however requires much more developing effort than a FTU process as a result of conflicting with other processes (Horizontal influence), or outside the coverage of existing SCMS infrastructure (Vertical influence). This definition implies that FTC and FTU are relative to each practice depending on the capabilities of the information system and its developers, and even the budget constraints.

FTC and FTU demand different levels of implementation strategy. The former mainly affects choices of system architecture and foundational technologies which have a principal contribution to the system initial cost. The latter characterises how easy it is to add or modify functions in the existing framework.

In total, the SCMS flexibility should aim to:

- Support most of the known critical processes;
- Enable system tuning / reconfiguration to accommodate varieties;
- Be extensive to align with strategical changes in its lifetime, and to cooperate with key customers / suppliers effectively;
- Add or update functions with minimal impact on the whole system.

The information industry is still at the infant stage as although computer hardware runs much faster than decades ago, the software techniques lag behind. For the time being, software developers usually need to compromise between flexibility, usability and overall cost as sheer flexible systems are extremely expensive to develop and maintain. Such an excessive flexible system also gives difficulties in training and daily use at the end user side. In addition, a commercial Management Information System (MIS) usually has a limited life cycle. It will be replaced by new versions which contain either better quality rework, new technologies or extra features. This is identified as a fundamental phenomenon of software products (Lehman, 2000). It is not realistic to plan a permanent and monolithic system that handles everything. Information managers are therefore expected to draw the boundary of system functional coverage within its lifetime, in order to align with business developing strategies and available finance resource. In short, a tangible survey includes questions of "what the right system should be?" and "what's the overall cost during its life time?" is vital to a successful SCMS practice. That is what this research tries to give answers to.

2.3 Supply Chain Performance Measurement

In terms of effective and flexible supply chain management, could existing performance measurement techniques be applied to estimate the overall SCMS cost? Research literature (Gunasekaran, Patel and Tirtiroglu, 2001; Chen and Paulraj, 2004) suggested performance measurement should be conducted to align with overall company visions and strategies. Neely et al. (1995) and Shepherd and Gunter (2005) summarised research initiatives on performance measurement methodologies and metrics comprehensively. According to their conclusions, insufficient attention has been paid to indicate the practical route towards an effective information infrastructure to boost overall supply chain performance, especially in the TMF integration segment.

A possible resolution to tackle the supply chain performance measurement problem could be a specific SCMS subset definition within the performance matrix, or an "optimal guideline" with respect to cost, capability, difficulty and flexibility. For example, Talluri (2000) presented a multi-objective model to evaluate information systems for SCM. His model integrated four critical performance indicators: flexibility, quality, time and cost. However, this empirical approach was even more complex than general supply chain performance measurements in the way it cross-referenced various business strategic objectives and Key Performance Indicators (KPIs) (Parmenter, 2007), apart from the problem that there was little evidence to prove its correctness.

In brief, performance matrix based approaches share the common disadvantages and difficulties in establishing the performance indicators. They also suffer from the lack of standards to collate the relevance between business performance changes and information system evolution. Besides, since there are so many performance measurement models announced, coupled with the fact that most of them are established on specific case studies, it is not clear that which one should be followed for the SCMS implementation scenario.

2.4 E-Commerce and Supply Chain Management

The Internet and E-Commerce has changed the supply chain management exercises dramatic in the way of information exchange and efficiency improvement (Johnson and Whang, 2002). Modern supply chain management systems usually comprise a set of E-Commerce processes as one of its core modules. In fact, modern supply chains are commonly recognised as E-Commerce enabled supply chains (Murillo, 2001; Disney et al., 2004). Apparently it should be a choice to apply research outcomes from the E-Commerce domain to the SCM.

However, according to Disney et al. (2004), the impact measurement of E-Commerce over a Supply Chain is mainly performed on a known strategy *after* implementation; there are few predictive elements to indicate the best practices in advance. It is close to impossible to compare E-Commerce system implementations directly due to the tremendous uncertainty and discrimination. Various methods, including statistic based, continuous and discrete control theory and simulation, have failed to indicate how to reduce cost in a practical way. Furthermore, an information system itself adds considerable complexity to manual decision making and calculation due to excessive information presented to the end users. It is too difficult to predict all the details of what will happen in advance. This conclusion was backed up by Gunasekaran and Ngai (2004) which stated that there were very few research initiatives or comprehensive frameworks that deal with E-Commerce system cost in the

SCM context.

2.5 Value-Based Software Engineering Cost Estimation

The objectives of this study still fall in the broad scope of software engineering. Naturally, it is expected the outcomes of existing software engineering researches to be applicable in the SCMS field. In the survey by Boehm, Abts and Chulani (2000), several classes of software cost estimation models were reviewed. It presented a sketch of the state-of-the-art techniques in this area and possibly a resolution for the SCMS cost estimation dilemma. The key findings in this survey were: firstly, all the cost estimation techniques were challenged by the rapid pace of changes in the software technology; and secondly, although each individual method had its strengths, there was no single one that was better overall than others. It was suggested by the authors to combine multiple methods with comprehensive cross-referencing. This kind of method combination however demands high level skills and experience from the practitioners. Boehm (2003) termed these kind of software enginering as Value-Based Software Engineering (VBSE). By definition, VBSE aims to develop models and measures to capture the software application value to assist the managers, developers and users for decision making between cost and quality, functionality and schedule, as such decisions must be economically feasible and comprehensible to the stakeholders with different value perspectives. In this section, a typical cost module - COCOMO, is investigated to clarify the feasibility of applying it for SCMS cost estimation.

COCOMO, short for COnstructive COst MOdel (Boehm, 1981), was a case-study based software cost estimation model first known to the public in the late 1970's and further refined as a software project planning tool for the Ada language[2] (Boehm and Royce, 1989). A major version upgrade to COCOMO II (Boehm, Steece and Madachy, 2000) was formulated recently to enable early stage prototyping and incremental development, in order to keep up with software technical evolutions such as object-oriented programming, reusable components, Rapid Application Development (RAD), and the like.

In the COCOMO II, a three-phase approach was proposed to obtain a cost estimation:

1. Application composition model at the earliest phase;
2. Architectural function points counting;
3. Fine-grained cost calculation based on several cost drivers as introduced in the original COCOMO.

The prime obstacle that hinders a successful application of COCOMO II in the supply chain environment lies in its substantial details to be determined: COCOMO II is not a definitive model that can be applied for an instant cost estimation by feeding values into some formulae. The final cost is still subject to revisions based on subsequent data analysis and proper balancing of object points (by counting forms, reports, data tables, etc.), function points, source code, programming language choice, composition systems, flexibility grade, and more. Without the system design detail and in-depth background information technologies, a rational cost estimation could not be established.

[2] Ada is a structured, statically typed, imperative and object-oriented high-level computer programming language, extended from Pascal and other languages. More details can be found in (Gehani, 1983).

Yet another assumption in the COCOMO-style approach which introduces deviation to the SCMS cost estimation is the progress of development. COCOMO regards any software project as a monolithic developing process, which is not suitable for the Supply Chain environment. Therefore, it is asserted that it might be possible to apply a partial COCOMO II model for a coarse-grain cost estimation in the SCMS context, but a simplified SCM specific cost model with respect to flexibility and uncertainty is more applicable for the majority of SCMS practitioners.

In recent years the incremental (iterative) programming has been popular in the software industry. Graham (1989) defined this developing method as a software production style carried out by a series of increments throughout the project life cycle. The developing procedure of SCMS is analogous to incremental (iterative) development in several ways. Both undertake phased development and both have certain degree of uncertainty on top of the stationary infrastructure. Therefore the research findings from incremental development might be valuable to the SCMS cost estimation topic.

While the benefits of incremental development in risk reduction and quality improvement are remarkable, its economic implications to the overall software project have not been well studied. Benediktsson et al. (Henry Ford, Mass Production, Modernism and Design) proposed a research initiative that attempted to apply COCOMO II cost estimation model to obtain the cost effect of the incremental developing pattern. To satisfy the prerequisites of COCOMO II, this research was coerced to compromise on several hypotheses and limitations:

1. Functional components are discrete from each other so that they can be implemented and delivered independently without affecting each other. The SCMS cost estimation model of this study is actually establish on

the same premise. Otherwise, one uncertain factor can trigger a chain reaction of changes, this makes it impossible to assess and quantify precisely before the actual occurrence. However, this research does attempt to consider this phenomenon by introducing the FTC process type at statistical level.

2. It assumes that the initial design overhead has a linear relation with the number of iterations but there is no sufficient evidence to underpin this correlation.

3. It also assumes that the *Breakage*[3] remains constant. In other words, the percentage of extra incremental rework is fixed. There is no statistical or theoretical evidence to support this hypothesis for real world projects, including the SCMSs.

4. According to the equations proposed, the final result is very sensitive to the degree of scale exponent E, but a clear guideline on how to pick the appropriate value does not exist.

5. The equations contain insufficient parameters to depict primary features of a SCMS project.

As the above conclusion suggests, the introduced method is by far an early stage prototype. There is still much work to be continued to make it applicable in real projects. The above hypotheses and limitations are not satisfying enough to be accepted as the SCMS cost estimation model.

[3] A term introduced in (Boehm, Steece and Madachy, 2000) to indicate the percentage of code rework

2.6 Gebauer and Schober's Research

Gebauer and Schober (2006) proposed a unique theoretical information system cost model. This cost model was established on an abstract high-level business process model with attention on process uncertainty and variability. On the basis of this process model, statistic abstraction was performed to deduce the cost model formulae. The final acquisition, which was close to the objectives of this study - a practical information system cost estimation model, was compromised by a few limitations. These limitations, in particular the time-irrelevance, require further research with regards to their adaptability in the supply chain context. Apart from these imperfections, its unique process model and statistic abstractions suggested a possible way towards the overall objectives of this research. In the following sections Gebauer and Schober's cost model will be studied in detail.

2.6.1 The cost model

According to Gebauer and Schober (2006), a business process is handled by either computer or human. In the case that a process request is already known but not yet implemented in computer software, the ordinary choice is to pursue a system upgrade either by extending the existing software (FTU) or by adding / replacing the whole function module (FTC), depending on the information system flexibility. Alternatively, this type of process can remain manual if appropriate. The same happens to those unknown processes apart from higher challenges to system flexibility. A certain proportion of process types might be left outside the information system coverage throughout by manual handling for various reasons (For instance, very low occurrence frequency or limitation of system flexibility). The total cost is summarised based on the occurrence count of each process route. Figure 2.1 illustrates the process handling flowchart.

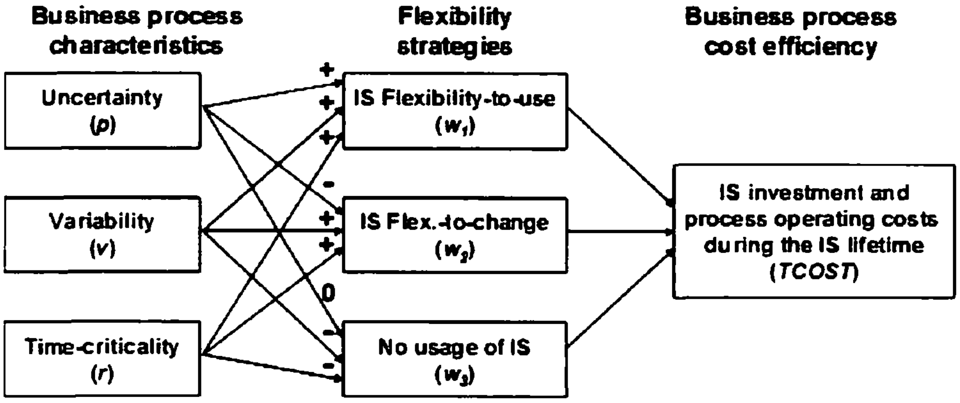


Figure 2.1: The generic process handling workflow by Gebauer and Schober (2006)

In Figure 2.1, a business process consists a set of tasks (sub-processes), the actual process occurrences can be completed via various optional tasks. The uncertainty(p) indicates the difficulty to predict the exact tasks and resources required. Therefore, the amount of tasks and their degree of uncertainties determine the overall uncertainty of that business process. Process variability(v) is the degree of process occurrences concentrating on certain tasks. It is considered high when task types are performed evenly. The time-criticality(r) in this context indicates the share of process occurrences which have to be performed at high priority. In most case, computer supported business processes can be handled more efficiently than manual work therefore time-criticality adds weight to the system implementation decision.

Based on these process characteristics, the calculation path to the total cost are connected by arrow lines. The "+" symbol in the diagram beside an arrow indicates the source characteristic introduces a high value to the target parameter, while the "-" symbol leads to a low value. The "0" symbol means the effect is unknown. A cost calculation model is therefore constructed from this cost flow chart.

The time-criticality(r) has been proven by the model proposers to have minor impacts to the final cost, and also, this factor is rarely distinguished in actual practices. In fact, most business processes are expected to be handled in time otherwise the organisation has run into problems. Business organisations usually employ extra workforce or upgrade the system whenever a performance bottleneck is identified. This time-criticality factor is therefore not taken into account in this temporal SCMS cost estimation model.

2.6.2 Process variability and statistical representation

Gebauer and Schober (2006) assumed that information software handled multiple types of processes, each of which held a different occurrence rate. To estimate the total cost, the statistical analysis called cumulative probability distribution was applied. Many factitious or natural stochastic phenomena, for example social incomes, word frequencies, census of population, are distributed subject to the law of probability distribution, or sometimes referred to as the POWER-LAW DISTRIBUTION. The PARETO DISTRIBUTION[4] was the first well-known mathematical representation introduced to describe the allocation of wealth, as it elaborated the phenomenon that a large portion of the wealth of any society was owned by a smaller percentage of people. This theory is often expressed as *the 80-20 rule*, which indicates that 20% of people owns 80% of the social wealth. This rule can be extended to many other stochastic areas: 20% of customers contribute 80% of the revenue; 20% of the

[4]Developed by Vilfredo Pareto (July 15, 1848 - August 19, 1923, Geneva), a French-Italian sociologist, economist and philosopher. He made several important contributions in those areas and helped developing the field of microeconomics.

documents contain 80% of information to be collected; and in the supply chain context, 80% of process occurrences fall in only 20% of process types.

There are many other cumulative probability distributions such as PIGOU (Pigou, 1912) and ZIPF (Gunther et al., 1996). Most of these variances lead to the same result and are mathematically transferable (Adamic, 2000; Reed, 2001). In the case of process handling, the distribution is a simply stochastic distribution which can be easily represented by any of them. Gebauer and Schober (2006) opted the LORENZ CURVE (Lorenz, 1905), an inverted variant of the PARETO DISTRIBUTION, to synthesise and measure the process variability in a mathematical approach.

The Lorenz Curve (GASTWIRTH, 1971) is a cumulative probability distribution diagram to represent the distribution degree, typically applied to characterise household income distributions (Figure 2.2). The line of perfect equality, as its name indicates, represents a perfect equal distribution to all families, while the line of complete inequality indicates the whole income is possessed by a single family. The regular distribution always falls in between, appeared as the red curve in the diagram.

A SCMS is essentially a multi-type stochastic process handling system, which is Lorenz Curve compliant by accumulating the shares of process occurrences represented on the Y-axis and the process type share represented on the horizontal X-axis. Under this circumstance, the curve will be steep (draws near to the line of perfect inequality) if most process occurrences concentrate on a small amount of process types and vice versa. For example, in the order process scenario, suppose a company receives orders from 10 customers in various formats / types. If 80% of the orders come from a single customer (that is to say, in a single type), it can be asserted that the variability is low. If there

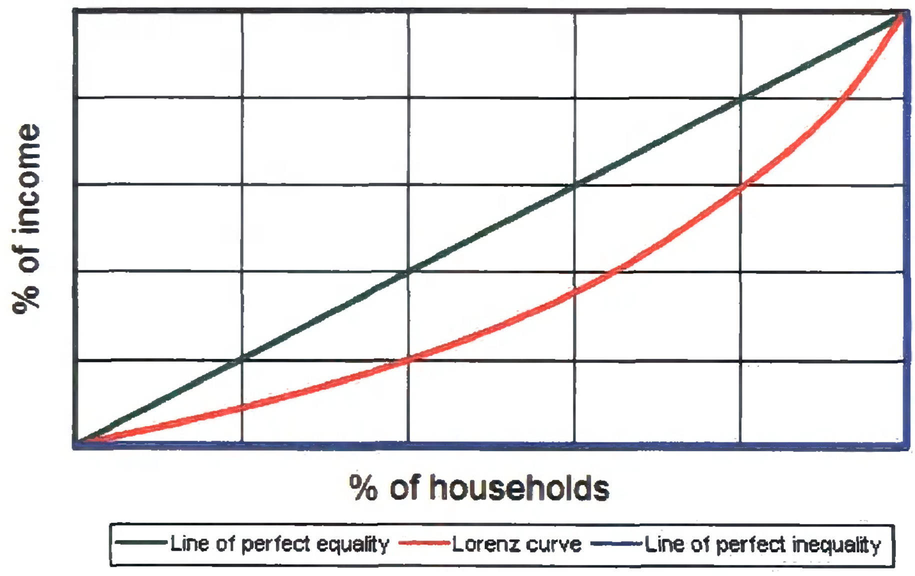


Figure 2.2: The LORENZ CURVE diagram

are no system developing activities after the initial release, the total cost can be easily estimated by summarising average process unit cost with occurrence count.

Recalling the objectives of the present study, the cost model should be established in a dynamic environment where quantity of FTU and FTC processes decrease gradually along with the continuous SCMS development. A different approach from Gebauer and Schober (2006) is required to utilise the Lorenz Curve. Under the premise that process occurrences arrive at even interval, the SCMS process developing sequence should be prioritised by the occurrence frequencies of process types to achieve cost saving, provided the unit manual handling costs are similar for each process type. Furthermore, it is cost effective to have SCMS functioning in a supply chain where the Lorenz Curve curvature is steep because more process occurrences are automated at a small developing cost. The strict relationship between the Lorenz Curve curvature and the process developing costs is described in Chapter 3 after introducing the single process scenario.

2.6.3 Model advantages

The major contribution of Gebauer and Schober (2006) is that they managed to establish a theoretical cost estimation model from a unique process handling workflow that concentrates on flexibility and variability. A theoretical cost models is more practical when comparing with empirical guidelines or questionnaire based methods (Byrd and Turner, 2000), especially for inexperienced users.

Another equally important key achievement originated in Gebauer and Schober's research was the application of the cumulative probability distribution - the Lorenz Curve, as mentioned earlier. It was introduced to synthesise the process variability in a quantitative approach. By means of this statistic abstraction, the necessity to consider occurrence frequency for individual process types was relieved by the decision of a single variable - the Lorenz Curve curvature. This abstraction greatly benefited the model simplicity, and hereby it is inherited in this research.

Within Gebauer and Schober's cost model, a certain degree of information system knowledge is still expected to specify reasonable values for the model variables introduced. Nevertheless the cost model is fairly practical as a whole, attributing to the well-defined variables and elimination of adjusting coefficients so that personal biases can be avoided.

2.6.4 Limitations and possible improvements

Given the fact that Gebauer and Schober's cost model was established on an "arbitrary software system", the lack of specific concerns to the Supply Chain characteristics makes it no more accurate than other cost estimation methods. A few additional shortcomings, as the side effects of its simplicity, are also identified.

Firstly, it can be asserted that information assets (both hardware and software) have limited lifetime. This lifetime should be taken into account at the planning stage to draw the boundary of flexibility and ultimately to achieve cost effectiveness. Gebauer and Schober's model neither defined the system lifetime explicitly nor allowed users to specify it in any way.

Secondly and more importantly, as this model was established on a relatively *static* system configuration. The deviation can be significant if applied to dynamic supply chains where system developing activities and requests for changes occur continuously.

Lastly, the time dimension has to be taken into account in the supply chain cost estimation since SCMS implementation is a continuous development over a long period. The point-of-time of each process development is not trivial to the final cost. The purpose of this research is to overcome these limitations.

Chapter 3

A Cost Estimation Model For SCMS

In this chapter, a SCMS cost estimation model is derived progressively, initiated from the single supply chain process model and inherited advantages and innovative ideas of Gebauer and Schober's research inherited.

3.1 Model Concept

Gebauer and Schober's abstraction of a process handling workflow (Figure 2.1) illumined a feasible starting point of the cost model construction. Regardless of whether the supply chain information system is made up of commercial components or in-house developed kits, it is safe to claim that a certain proportion of the system will go through a reconfiguration or developing cycle during its life time to accommodate changes from the host supply chain. Therefore, this model asserts the sub cost of arbitrary unit supply chain process comprises of:

- initial cost
- manual process cost
- software developing cost
- system process cost

The source of initial cost could be from computer hardware or software purchasing, consultation as well as various labour charges. Essentially this cost represents the total amount of the one-off costs to establish the system infrastructure. In reality, it is up to the decision makers to balance the share of cost which could also be used for other purposes. For instance, computer server hardware purchased to host the SCMS can also be used by other applications to exploit its full capacity. In this case the share of SCMS hardware cost is basically a financial trade-off. As this kind of cost is usually countable at the early stage, it is simply denoted as a single cost element in this cost model so as to focus on the growth of dynamic costs. Unit process developing cost is treated as a lump sum for simplicity as no matter how the actual cost occurs, the one-off sub-total remains invariant. Manual handling cost is regarded as a linear function to process occurrence count, multiplied by a fixed unit manual handling cost per occurrence. That is to say, unit manual handling cost is regarded as invariant for all process types. Analogously, unit system process cost of any process type is also treated as an invariable.

3.2 Single Supply Chain Process Cost Analysis

The cost estimation model construction starts from inspecting the cost trend of a single generic supply chain process scenario, as similar to Gebauer and

Schober's research. In this scenario, if the process type is new to the SCMS, the process requests will be handled manually for the time being. Meanwhile, software developers collect technical details in order to automate this process type. The actual program is developed once the design is confirmed. These SCMS developing activities should be associated with a certain amount of one-off cost. During this period, transactions of this process type are still handled manually before the developing work finishes. Attribute to the replacement of manual handling with system automation, the cost growth scales down to a lower level from that time-point until the end of the system life time. This temporal cost transition is clarified better in line-chart style as Figure 3.1.

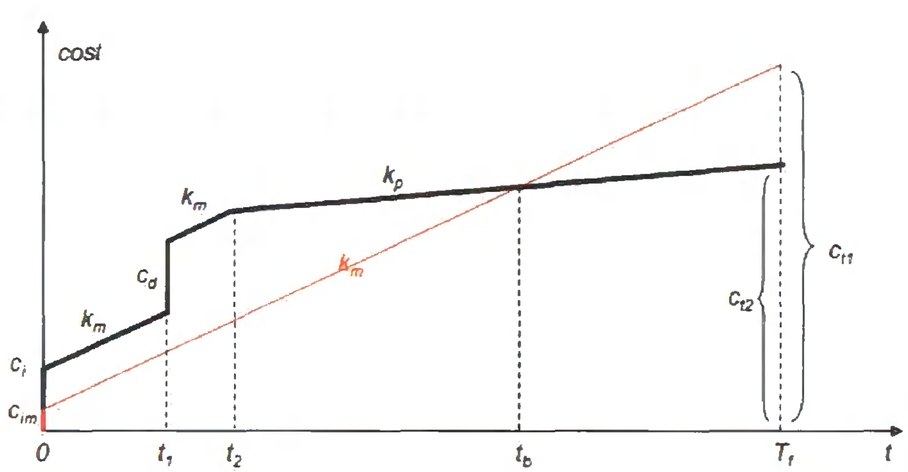


Figure 3.1: Single process cost transition line chart

In Figure 3.1, the bold black broken line outlines the estimated sub total cost for this process to the end of the system life time T_f. c_i stands for the initial cost and c_d indicates the developing cost, both are one-off cost. k_m and k_p are slopes that denote the cost spent per unit time via manual handling and system automation respectively, therefore it is asserted that $k_p < k_m$ (Otherwise it is more cost effective to leave the process type to manual work). The t_1 and t_2 represent the starting and ending time points of process development.

The red broken line drawn in Figure 3.1 is the pure manual process cost line provided for comparison. It is composed by the initial manual setup cost c_{im} (which comes from various occasions such as employment or procedure establishment) as well as the running cost at slope of k_m. After the break-even point t_b, the overall sub total c_{t2} goes lower than the pure manual sub total c_{t1}. This line chart also implies that the manual operation is more cost effective when $T_f \leq t_b$. This could be one of the reference points to decide whether a certain process type should be implemented in a SCMS.

The mathematical representation to Figure 3.1 is as follows:

$$c_{t1} = c_{im} + k_m \cdot T_f \tag{3.1}$$

$$\begin{aligned} c_{t2} &= c_i + k_m \cdot t_1 + c_d + k_m \cdot (t_2 - t_1) + k_p \cdot (T_f - t_2) \\ &= c_i + c_d + k_m \cdot t_2 + k_p \cdot (T_f - t_2) \end{aligned} \tag{3.2}$$

where $k_p < k_m$.

Before looking into the compound multi-process scenario, two key points have to be addressed:

1. The introduction of cost slopes of k_m and k_p implies the assumption that each process type has a steady process transaction density. In reality, process occurrence frequencies may vary over the time. For example, it

could be the case that a company received a large volume of orders from a supply chain partner in the previous quarter but just a few for the current quarter due to competition from a cheaper supplier of the same product. This kind of stochastic transitions can hardly be estimated in advance and expressed in any cost estimation model accurately, therefore the referenced k_m and k_p should be regarded as average values. This approximation will both facilitate the value estimation and the cost model itself although deviations is involved inevitably.

2. In the single process scenario, which uncertainty category the process falls into does not need to be distinguished. The significant difference between these two types is the software developing cost c_d. An uncertain process to the SCMS will introduce a larger c_d. This difference will be taken into account in the compound process model.

3.3 Multi-Process Cost Composition

A supply chain management system usually supports multiple SC processes. Regardless of how the system is constructed, some of the most common processes should be implemented at the beginning of system implementation. Considering the high uncertainty of a supply chain, a large proportion of processes will be developed as upgrades to the existing SCMS on a sequential basis in accordance with business process adjustments or supply chain environmental transitions.

As a fundamental difference to the cost estimation model of Gebauer and Schober (2006), this cost model treats the developing procedure in a sequential fashion with respect to system life time rather than a plain cost summation of all process types (namely, the parallel developing mode). This is due to the consideration that in reality, no matter whether the supply chain system is implemented in-house or by contracting to software vendors, the SCMS working group maintains a steady size and productivity most of the time. A stable developing group is the standard setup recommended in much software engineering research literature (Brooks, 1975; Sommerville, 2000, chapter 22) for effectiveness and risk control. With these external constraints and possible process inter-dependencies, this cost model opts the sequential developing mode as the foundation of process combination to formulate the cost estimation model.

It is also necessary to indicate that each process type has its own process occurrence rate. The total cost can be reduced by prioritising the most frequent process types so that more tasks can be handled by computer in total during the lifetime of SCMS. This cost model assumes SCMS development is organised in this optimal sequence. In reality, there might be exceptions that override this priority order, such as tactical decisions, or wrong developing arrangements due to insufficient information. The cost model however cannot capture these stochastic disorders so it considers the ideal condition only which leads to the minimal cost. A disordered SCMS implementation indicates that the project is not a good practice as it violates the cost saving target. None of the cost estimation models will be able to give a reasonable estimation in this case.

The conceptual multi-process composite cost chart is illustrated in Figure 3.2. It indicates if the corresponding process type has already been supported by SCMS, then its tasks are handled by the system, or otherwise by manual work. Suppose all process occurrences distribute evenly (i.e. purely random), the following equations can be established accordingly:

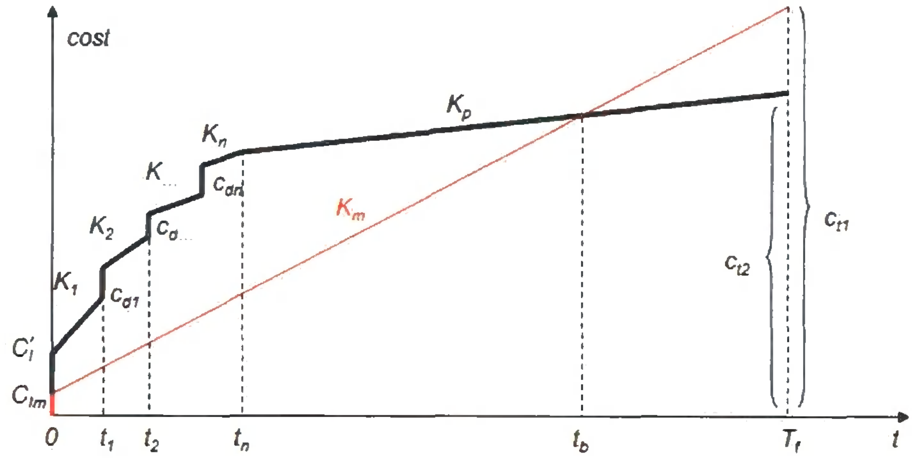


Figure 3.2: Multi-process composite cost line chart

$$C_{t1} = C_{Im} + K_m \cdot T_f \tag{3.3}$$

where C_{Im} is the blanket initial manual setup cost and $K_m = \sum_{i=1}^{n} k_{mi}$, n is the number of total process types.

$$\begin{aligned} C_{t2} &= C_I' + (c_{d1} + c_{d2} + \cdots + c_{dn}) + K_1 \cdot t_1 + K_2 \cdot (t_2 - t_1) \\ &\quad + K_3 \cdot (t_3 - t_2) + \cdots + K_p \cdot (T_f - t_n) \end{aligned} \tag{3.4}$$

where arbitrary K (e.g. $K_1, K_2, \cdots, K_p$) at sequence x can be expressed as $K_x = \sum_{i=1}^{x-1} k_{pi} + \sum_{i=x}^{n} k_{mi}$, provided the process development is *continuous*.

C_{t1} in Equation 3.3 is the total manual cost without a SCMS. This variable is introduced for reference purpose only to compare with the real total cost C_{t2} (Equation 3.4), which is this cost estimation model tries to capture.

The k_{pi} and k_{mi} denote process costs per unit time for type i that are handled by the information system and by manual handling respectively. c_{di} stands for system developing cost for process type i. By substituting K_x into the Equation 3.4, the following equation is derived:

$$\begin{aligned} C_{t2} = \; & C'_I + (c_{d1} + c_{d2} + \cdots + c_{dn}) + (k_{m1} + k_{m2} + \cdots + k_{mn}) \cdot t_1 \\ & + (k_{p1} + k_{p2} + \cdots + k_{pn}) \cdot (T_f - t_n) + (k_{p1} + k_{m2} + \cdots + k_{mn}) \\ & \cdot (t_2 - t_1) + (k_{p1} + k_{p2} + k_{m3} + k_{m4} + \cdots + k_{mn}) \\ & \cdot (t_3 - t_2) + \cdots \end{aligned} \tag{3.5}$$

Which can be simplified by merging the homogeneous items:

$$\begin{aligned} C_{t2} = \; & C'_I + (c_{d1} + c_{d2} + \cdots + c_{dn}) + (k_{p1} + k_{p2} + \cdots + k_{pn}) \cdot T_f \\ & + \sum_{i=1}^{n} (k_{mi} - k_{pi}) \cdot t_i \end{aligned} \tag{3.6}$$

In addition to the cost elements in Equation 3.6, there will be a number of processes supported in the beginning ($t = 0$). The cost associated to this pre-development is included in the C'_I of Equation 3.6 but worth being marked as a separate cost element C_S. The rest of all one-off initial costs are subsumed to cost element C_I , which are not process development related such as purchasing of hardware, software, consultation, etc.

$$C'_I = C_I + C_S \tag{3.7}$$

$$C_S = K_S \cdot T_f \tag{3.8}$$

where $K_S = k_{sa} + k_{sb} + k_{sc} + \cdots$. The k_{sa}, k_{sb}, k_{sc}, $\cdots$ stand for process cost slopes of arbitrary process types developed before system functioning.

On the other hand, some processes are never developed in the SCMS within the whole T_f period as a result of their low occurrence rates. These manual transactions incur the cost of:

$$C_M = K_M \cdot T_f \tag{3.9}$$

where $K_M = k_{ma} + k_{mb} + k_{mc} + \cdots$. The k_{ma}, k_{mb}, k_{mc}, $\cdots$ stand for process cost slopes of arbitrary pure manual process types respectively.

Finally, the total cost is summed up as:

$$C = C_M + C_S + C_{t2} \tag{3.10}$$

3.4 Model Refinement

3.4.1 Lorenz Curve representation

Although Equation 3.10 does give the actual total cost, it is hardly useful for early stage cost estimation when most of the process details are unknown. To refine this model, it is essential to understand that process types and their occurrence rates are fundamentally different notions: A process type introduces one-off developing cost c_{di}, while a process occurrence rate contributes to the day-to-day operational cost, which is reflected by k_{pi} and k_{mi} in the equations. Process occurrence rate is also the primary indicator to suggest whether the process type should be implemented by a SCMS as well as its developing priority. This cost model will apply the Lorenz Curve to map the relation of these two aspects due to its similarity to the population - income share proposition that Lorenz attempted to characterise (Lorenz, 1905). Applied to this model, the relation can be denoted as:

$$y = L(w, v)$$

This expression indicates that w proportion of process types share y percentage of total process occurrences that will happen in the designated SCMS life time, with the process types arranged in the incremental frequency order (as per Lorenz Curve prerequisite). The Lorenz Curve curvature v indicates the degree of process type distribution.

Specifically, in order to achieve the closest match from application of Lorenz

Curve against actual cost, it is assumed that the actual system development follows the underlying optimal rules (visualised in Figure 3.3):

1. Process types with the highest occurrence rates (share of w_1) are implemented at the system initial stage.
2. Medium frequency process types are developed after the system functioning in the order of their occurrence rates. This proportion of share is denoted as w_2.
3. Process types with the lowest occurrence rates are left to manual handling. $1 - w_1 - w_2$ represents this proportion.

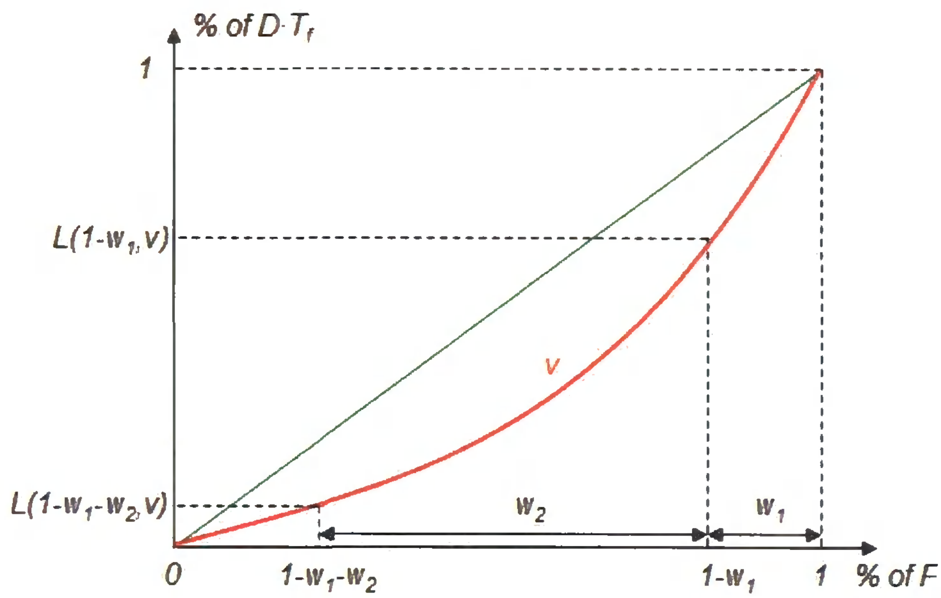


Figure 3.3: Lorenz Curve representation of the relation between process types and their occurrence rates

The initial Lorenz Curve was introduced as a visual graph to present the probability distribution proposition. Various mathematical representation models have been initiated over the years (Kakwani and Podder, 1976; Basmann et al., 1990; Ortega et al., 1991). However the best fit to the actual curve is still arguable. For the intention of approximate cost estimation, to facilitate future work, a single-variable model proposed by Chotikapanich (1993) is opted for this cost model. Equation 3.11 is the form of this model.

$$y = L(w, v) = \frac{e^{v \cdot w} - 1}{e^{v} - 1} \tag{3.11}$$

where curvature $v \in (0, \infty)$.

This mathematical model has its limitations because neither perfect distribution nor perfect inequality can be reached. Fortunately, for the supply chain process handling topic, these extreme conditions do not likely occur. In the following chapters, the curvature range is artificially bounded to $v \in [0.01, 20]$ as the practical scope. Figure 3.4 demonstrates Chotikapanich's $L(w, v)$ plotted at various curvatures. This diagram shows the curvature range can cover most of the result space therefore it is sufficient for the cost model scenario.

3.4.2 Lorenz Curve based model abstraction

To provide a practical cost estimation model by means of the Lorenz Curve, each cost element will be reviewed to compose the simplified form. Prior to further mathematical refinement, a few extra model variables have to be introduced.

1. In Equation 3.6, as process occurrences are discrete, arbitrary k_{pi} and k_{mi} comply to the forms of $k_{pi} = \mu_{pi} \cdot f_i$ and $k_{mi} = \mu_{mi} \cdot f_i$, where μ_i

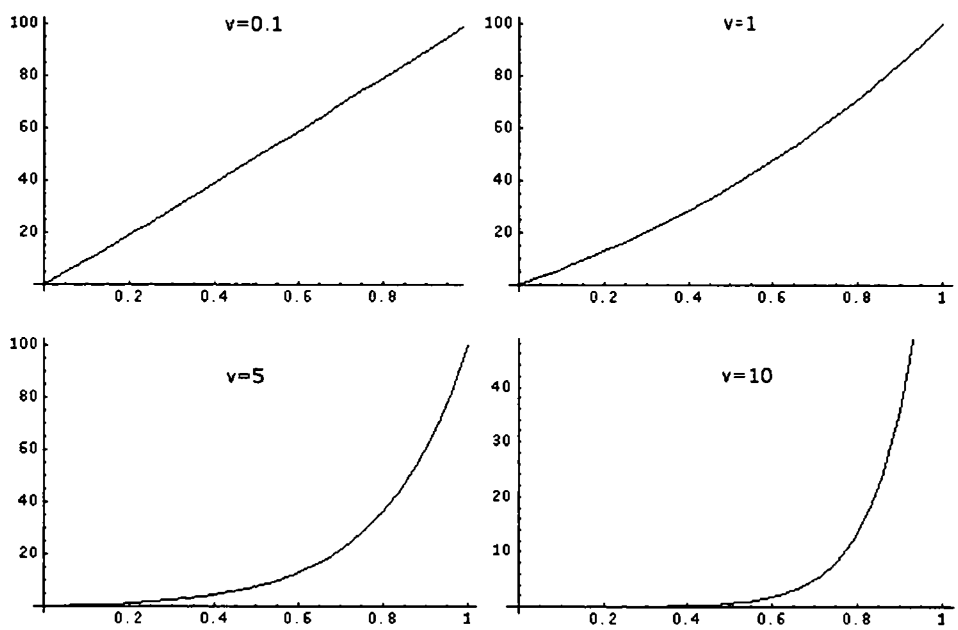


Figure 3.4: Chotikapanich's Lorenz Curve representation at various curvatures

represents unit cost of a single occurrence and f_i denotes for occurrence share of process type i.

2. F is defined as the count of total process types.

3. It is assumed process transaction density remains steady in the main (as explained in the single process model). Therefore the variable D is introduced for total process occurrence count per unit time. Consequently, the total occurrence count in system life-time can be marked as $D \cdot T_f$.

4. It is also presumed that proportion x of process types fit into the flexible-to-change category in the group of processes that are implemented after SCMS functioning (namely within $F \cdot w_2$).

5. Software development work introduces a lump-sum average unit cost dc to each FTU process, and likewise a separate dc' to each FTC process.

From the above definitions in conjunction with Figure 3.3, C_M and C_S can be amended to:

$$\begin{aligned} C_M &= K_M \cdot T_f \\ &= L\,(1 - w_1 - w_2, v) \cdot D \cdot T_f \cdot \mu_m \end{aligned} \tag{3.12}$$

$$\begin{aligned} C_S &= K_S \cdot T_f \\ &= [1 - L\,(1 - w_1, v)] \cdot D \cdot T_f \cdot \mu_p \end{aligned} \tag{3.13}$$

The system developing cost $(C_{d1} + C_{d2} + \cdots + C_{dn})$ in Equation 3.6 can be defined separately as cost element C_D. If the initial process development is considered as part of C_D (unit process type developing cost is dc), then this cost element can be denoted as:

$$\begin{aligned} C_D &= \sum_{i=1}^{F\cdot(w_1+w_2)} c_{di} \\ &= F \cdot w_2 \cdot x \cdot dc' + F \cdot [w_1 + w_2 \cdot (1 - x)] \cdot dc \end{aligned} \tag{3.14}$$

In the same way, system operational cost element $(k_{p1} + k_{p2} + \cdots + k_{pn}) \cdot T_f$ in Equation 3.6 is defined as C_R:

$$C_R = (k_{p1} + k_{p2} + \cdots + k_{pn}) \cdot T_f$$

$$
\begin{aligned}
&= \sum_{i=1}^{F\cdot(w_1+w_2)} k_{pi} \cdot T_f \\
&= \left[1 - L\left(1 - w_1 - w_2, v\right)\right] \cdot D \cdot \mu_p \cdot T_f \qquad (3.15)
\end{aligned}
$$

The last cost element comes from the developing stage where manual handling is gradually replaced by system upgrade. This cost element is denoted as C_X:

$$
C_X = \sum_{i=1}^{F\cdot w_2} \left(k_{mi} - k_{pi}\right) \cdot t_i \qquad (3.16)
$$

The argument t_i in Equation 3.16 represents the time gap from system start time $t = 0$ to the point that process type i has its turn to be developed. The precise t_i can only be ascertained after t_1, t_2, ..., t_{i-1} are known. These details are obviously not available at the planning stage, before their actual occurrence. However, since the *average* developing costs dc and dc' have already been utilised, it is viable to define the average developing durations τ and τ' under the premise that the developing team maintains a steady productivity. This is because when equal costs indicate same code sizes, they also lead to equal developing durations if the team productivity maintains steady (Boehm, 1981; Sommerville, 2000).

Under the assumption that this model prioritises the developing order by process occurrence rate to comply with the Lorenz Curve prerequisites, it is unknown to the cost model for whether a process in the $F \cdot w_2$ scope (i.e., after system functioning $0 < t_i < T_f$) falls into FTU or FTC category. This uncertainty prevents us from applying appropriate τ to individual process types, therefore τ and τ' have to be approximated to the same value. In the mathematical representation, it indicates:

$$\tau = \tau' = (t_i - t_{i-1}) \tag{3.17}$$

where $t_i = i \cdot \tau$

This assertion undoubtedly involves deviation to the total estimated cost. A possible work-around might be to increase the value of dc'. Nevertheless, there are no definitive answers to the questions of how much to adjust or what are the side-effects on the final cost so far. This is worth being researched as future work.

Based on the above analysis and simplification, C_X could be reformed to:

$$C_X = \sum_{i=1}^{F\cdot w_2} (\mu_m - \mu_p) \cdot f_i \cdot \tau \cdot i \tag{3.18}$$

By applying the Lorenz Curve to C_X , the following equations are tenable:

$$\begin{aligned} C_X = & \sum_{i=1}^{F\cdot w_2} \left[L\left(1 - w_1 - \frac{i-1}{F}, v\right) - L\left(1 - w_1 - \frac{i}{F}, v\right)\right] \\ & \cdot D \cdot (\mu_m - \mu_p) \cdot \tau \cdot i \end{aligned} \tag{3.19}$$

$$\begin{aligned} C_X = & D \cdot (\mu_m - \mu_p) \cdot \left(1 + e^{-\frac{v}{F}} + e^{-\frac{2v}{F}} + \cdots + e^{-\frac{(F\cdot w_2 - 1)\cdot v}{F}} - F \cdot w_2 \cdot e^{-v\cdot w_2}\right) \\ & \cdot \tau \cdot \frac{e^{v\cdot(1-w_1)}}{e^v - 1} \end{aligned} \tag{3.20}$$

According to the finite geometric progression law, it can be proved that:

$$1 + q + q^2 + q^3 + \cdots + q^n = \frac{1 - q^{n+1}}{1 - q}$$

Let $q = e^{-\frac{v}{F}}$ and $n = F \cdot w_2 - 1$, the last quoted proportion in Equation 3.20 can be transformed to:

$$1 + e^{-\frac{v}{F}} + e^{-\frac{2v}{F}} + \cdots + e^{-\frac{(F \cdot w_2 - 1) \cdot v}{F}} = \frac{1 - \left(e^{-\frac{v}{F}}\right)^{F \cdot w_2}}{1 - e^{-\frac{v}{F}}} = \frac{1 - e^{-v \cdot w_2}}{1 - e^{-\frac{v}{F}}}$$

, whereby

$$\begin{aligned} C_X &= D \cdot (\mu_m - \mu_p) \cdot \left(\frac{1 - e^{-v \cdot w_2}}{1 - e^{-\frac{v}{F}}} - F \cdot w_2 \cdot e^{-v \cdot w_2} \right) \\ & \cdot \tau \cdot \frac{e^{v \cdot (1 - w_1)}}{e^v - 1} \end{aligned} \tag{3.21}$$

At last, the total cost sums up all these cost elements:

$$C = C_I + C_M + C_S + C_D + C_R + C_X \tag{3.22}$$

Equation 3.22 is the final form of this temporal cost estimation model. Variables referenced are summarised in Table 3.1. Table 3.2 enumerates the cost elements that are introduced during model derivation. They will be investigated in more details for the purpose of cost reduction in the following chapters.

Table 3.1: Cost model variable notations

Variable	Description
μ_m	Average unit cost per manual process occurrence
μ_p	Average unit cost per SCMS supported process occurrence
w_1	Process type share that implemented before system functioning
w_2	Process type share that implemented after system functioning
x	Share of FTC process types (within $F \cdot w_2$)
D	Total process occurrence frequency (i.e. count of process occurrences per unit time)
v	Lorenz Curve curvature to indicate process type-occurrence distribution degree
T_f	Planned SCMS life-time
F	Count of total process types
dc	Average flexible-to-use process developing unit cost
dc'	Average flexible-to-change process developing unit cost
τ	Average unit process developing duration

Table 3.2: Cost element notations

Cost Element	Description
C_I	One-off initial cost to establish infrastructure, developing team, preparation training, etc.
C_M	Manual process handling subtotal
C_S	Early stage developing cost before system functioning
C_D	System developing cost after system functioning
C_R	Supply chain management system operational cost
C_X	Mixed cost for the stage that manual handling is gradually replaced by system upgrade after system functioning

Chapter 4

Model Variable Sensitivity Analyses

Sensitivity analyses are performed on the cost model to identify the most cost-sensitive variables with their interaction effects. A few important variables are also examined respectively to investigate their influence to the final cost.

4.1 Introduction of Sensitivity Analysis

According to the proposed SCMS cost estimation model, the way to achieve a cost estimation appears straightforward in the first place: specify values of the variables in Table 3.1, together with the system initial cost C_I, and finish by feeding them into the model equations to get the final output C. The problem is that, as an "estimation" model for the dynamic supply chain environment, some variables of the cost model are very difficult to predict accurately in the real world. Also, as stated in the model construction section, some variables are dynamic or random in reality but normalised to be possibly included in this cost model. It is predictable that the model users would experience difficulties in determining the suitable values and have to estimate some of them by experience.

Cost deviations are consequently inevitable. Because of this, it would be useful to understand the cost impact of each model variable, so that the important ones can be treated with greater caution. For different supply chain environments the list of important or cost-sensitive variables could be vary, therefore the attention should be paid to the methods of sensitivity analysis rather than the experiment results. This chapter uses a set of experiments to identify these cost sensitive variables and to quantify their influences to the model output, in order to facilitate effective SCMS cost estimations.

In models involving many input variables, the Sensitivity Analysis (SA) methods (Cacuci, 2003) are developed to look at the effect of varying the inputs of a mathematical model on the output of itself, therefore to measure the sensitivity of model variables. Sensitivity analysis also provides a way to determine what level of accuracy is necessary for a variable to make the model sufficiently valid and useful. If a test result reveals that the model is insensitive to a certain parameter value change then it is safe to use a rough estimation for that parameter, instead of a value with greater precision, which could be very difficult to obtain. In some circumstances, sensitivity analysis may also be able to indicate reversely which parameter values are reasonable to use in the model, when the range of output has already been known. There exists several possible methods to perform the sensitivity analysis such as sampling-based analysis (Helton et al., 2006), Bayesian-emulator based analysis (Oakley and O'Hagan, 2004), variance-based methods (Saltelli et al., 2004), etc. Each of which has its respective strengths, weaknesses and conditions of use.

Apart from the mathematical domain where the sensitivity analysis is originated, similar problem settings have been noticed and studied in several research areas such as the Design Of Experiments (DOE) (Fisher, 1935), which is developed primarily within the domain of engineering statistics. In sensitivity analysis, one looks at the effect by varying the inputs of a mathematical model on the output. While with the DOE, which targets to generic real world processes, doesn't insist a mathematical model so long as the process has controllable inputs and measurable outputs.

The DOE theory is established on a powerful assumption that all inputs might have interactions with all other inputs (Barrentine, 1999) (which is to say, the inputs have influence to each other and ultimately changes the outputs in an indirect way that is hard to discriminate). In reality, due to the complexity or difficulty to carry out full set of experiments, not all DOE exercises look at every possible combination of factors and factor levels, they are called "Fractional factorial Design Of Experiments", while those do are called "Full Factorial Design Of Experiments". A full factorial DOE is practical only when fewer than 5 factors are being investigated. Testing the full combinations of more than 5 factors becomes over expensive and time-consuming.

In this model sensitivity analysis, the 13 model variables will be grouped into 2 different categories in order to achieve cost saving by adjusting the SCMS implementation rather than changing the capability of the system. In general, Chapter 4 performs the overall cost model sensitivity analyses where Chapter 5 pays attention to the system implementation only.

4.2 Value Ranges of Model Parameters

The proposed SCMS cost estimation model has complex non-linear relations to the set of variables, therefore the effects of variables to the model output could vary at different value levels. This suggests that the reasonable variable input ranges are to be determined in advance so that the outputs will be meaningful. With this restriction, the amount of necessary experiments is greatly reduced. More important, under this condition the model should generate outputs that are realistic as the pure mathematical model could lead to conclusions based on conditions that never exist. In the following sections, the 13 model variables will be reviewed in turn to establish a value range matrix as the foundation of the commercing analyses. Each variable is attributed to the type of either Control Factor or Environmental Factor, where a Control Factor can be altered directly (such as the speed of a belt, incline angle of a welding process, etc.) but an Environmental Factor complies with its supply chain environment only. Some of the SCMS cost model Control Factors are subject to various real world constraints but changeable indirectly to limited extents.

T_f - Planned SCMS lifetime

> A software system tends to have a limited lifetime T_f due to the fast pace of information technology evolution. After a certain period of time, the changes from dependent technologies and SC processes make it no longer economically worthwhile to maintain the system. Information technologies evolve so quickly that users are almost forced to give up old systems in order to get benefits from new technologies or sometimes only to keep the maintenance cost low. Although there is a lack of statistic about average enterprise information system lifetime, according

to typical industrial software release interval and servicing strategy, a 5 year duration is appropriate as the normal SCMS lifetime. To be comparable with manual processing, the 5 year lifetime is converted to hours on the basis of 52 weeks a year and 40 hours per week. This is to say, the SCMS lifetime is about $10,000$ hours. T_f is treated as an environmental factor accordingly.

F - Count of process types

The term "process" in the current context is recognised from SCMS point of view. This is to say, as long as two physical processes can be handled by the same SCMS program routine, they should be recognised as one process type. On the contrary, processes require seperate programming are of different types even they serve the same business purpose. For example, imagining a customer publishes orders via either Internet or traditional posting, even though the content information is exactly the same, two separate SCMS program routines (one downloads and converts information via Internet, the other scans the hard copy to get the information) have to be developed to handle the information. In this case, they should be distinguished as two different process types.

For the model analysis experiment, the accumulative process type range F of 100 to 200 (includes the re-development caused by supply chain changes) is assumed to represent a small size supply chain. F is determined by its supply chain business environment but not the SCMS, so it is clearly an environmental factor.

D - Total process occurrence frequency (count of process occurrences per unit time)

This variable indicates how many process tasks occur every unit time on average. For a small supply chain, it is reasonable to have about 20 to 30 processes every hour. A small team should be able to handle this amount of tasks without much pressure. An automated SCMS can obviously process much larger volume of information but the process frequency is driven by business demands but not the data processing capacity. From this point, variable D is an environmental factor. A SCMS does provide the flexibility of accommodating business activity increment (higher D value) with little influence to the final cost which is beneficial particularly in large scale supply chains.

τ - Average unit process developing duration

According to the definition of "process type", although it still depends on many other factors such as clarity of specification and the developing team, a normal process development should take about 2 to 3 weeks to complete. That is to say, 80 to 120 effective working hours.

The value of τ may be altered by hiring different skill level developers, by changing developing team size, or by using extra technical sources. It is a control factor to a SCMS project.

μ_m - Average unit cost per manual process occurrence

This variable indicates the unit cost to perform a single process without using the supply chain management system. The value could vary vastly due to many causes such as country and location, currency, salary rate, size of team, efficiency, etc. Instead of observing the absolute cost of a single process, the more practical way to achieve the average value is by calculating the cumulative cost over a period of time against the occurrence count. If a personnel is involved in supply chain tasks on a temporary or part-time basis, then only the pro rata cost should be accounted.

As a result, the cost model analysis has to be associated to a specific physical location to have consistent inputs and realistic outputs. The North East of England, where this research is based, is used as the default location for the sensitivity analyses. For a small size supply chain management team, it is assumed the team consists of a mixture of 10 full time employees from the business section, whose normal salary ranges from £15,000 up to £25,000 per year according to the statistic of Jobcentre+[1] (Year 2009), and also 2 employees from the finance section are involved in the supply chain operations by sharing 50% of their working hours. Their normal full time salary could be between £20,000 and £25,000 in this location. Finally, there should be a supply chain manager to coordinate the whole team. The usual salary for this type of role, according to the same information source, is about £35,000 to £45,000 every year, depends on level of skills and experiences.

The job cost is usually comprised of direct costs (for example recruitment cost, salary and pension / benefits, other costs such as travel expense or permits) as well as overhead costs (includes office rent, equipments, interest paid, other administrative costs). A normal practice to obtain the approximate job cost is by multiplying the base salary with a overhead coefficient. This coefficient can be deduced adversely from actual job cost against the salary for the particular organisation. In this research, coefficient of 1.4 is use to achieve the overall job cost. Therefore, the cost spent on salary every year should be in the range of £287,000 to £448,000 approximately. Average to a single process occurrence, μ_m falls

[1]Jobcentre Plus is an official job agency of the British government. Website http://www.jobcentreplus.gov.uk

into the range of 1.15 to 1.80.

Apart from the physical location, μ_m can also be influenced by many causes such as team size, skill level, efficiency and practicality of job routines, and so on. As the complement of an integral SCMS, μ_m should treated as a control factor.

μ_p - Average unit cost per SCMS supported process occurrence

For supply chain processes handled by computers, the running cost is sourced from information system maintenance as well as management overheads. For the above small supply chain setup, it should be enough to have an information system administrator who spends about 30% of his working hours on the SCMS and a dedicated supply chain manager (who has salary rate of £35, 000 to £45, 000, same as the manual setting.) to take care the rest of everything. According to the Jobcentre+, such an system administrator's normal salary is circa £30, 000 to £35, 000 a year. The total annual cost spent on SCMS supported supply chain, after taking the overhead coefficient 1.4 into account, should be between £61, 600 and £77, 700. Therefore the range of μ_p is about 0.25 to 0.31. To the very details there should also be some other trivial costs from broadband contract, hardware maintenance and replacement, technical training and software maintenance, disaster recovery plan, etc. Accordingly, the μ_p should be inflated a little to the range of 0.28 up to 0.34. In terms of variable type, μ_p is a control factor.

v - Lorenz Curve curvature

The Lorenz Curve curvature v is a statistic abstraction. To find out the value of a particular supply chain, it is necessary to perform statistic random sampling on the target supply chain to match the Chotikapanich's Lorenz Curve diagram (for example Figure 3.4) generated by Equation 3.11. To represent a normal supply chain, the curvature range of 3 to 6 is used by the following analyses.

In the SCMS context, the Lorenz Curve curvature v is the distribution balance degree of process types, which is purely business driven. It is an environmental factor.

w_1 - Process type share that implemented before the system functioning

In principle, the larger w_1 is, the more cost effective the SCMS will be. However, there are too many uncertainties in a supply chain to prevent a maximal share. Time constraint, insufficient information, constant changes, software flexibility are examples that lead to a small rate. SCMSs are different from traditional enterprise applications such as Enterprise Resource Planning system (ERP) where users can try to adapt to the system - SCMSs have to adapt to their external environment. Because of this reason, SCMSs are mostly bespoke systems. The realistic value range of w_1 is probably between 0.2 and 0.4. w_1 is a control factor since it is driven by system structure and early planning.

w_2 - Process type share that implemented after the system functioning

Considering a small portion of process types are never developed due to their low occurrence frequency, if this manual process type share is 0.1 to 0.15 (influenced by the Lorenz Curve curvature v and the break-even comparison of single process cost model in Chapter 3), the range of w_2 is therefore between 0.45 and 0.7. w_2 is a control factor determined by system capability and developing resources.

x - Share of FTC processes

This variable is the flexibility indicator of the system architecture. A well-designed open architecture may keep the FTC share x low by accommodating unplanned process type at low cost with reusable components. A sealed system (whose technical details are unknown to the users), even though very well designed, may not be a good choice since it prevents inheritance, integration and extension which is essential to FTU processes. The degree of x varies relying on the choice of implementation so an even range of 0.3 to 0.7 is used. The x is a control factor accordingly.

dc - Average flexible-to-use process developing unit cost

The unit process developing cost dc is another very implementation-dependent variable. The value range is widely scattered for different locations or different type of implementation. Fortunately, when implementing a real SCMS, dc should be measurable as most of the factors can be determined by the time. For the sensitivity experiments, it is assumed that the process development is done by contract developers which is typical for a small scale supply chain implementation. The salary for an experienced software developer in the North East England is about £$35,000$ to £$40,000$ per year according to the Jobcenter+ average salary. Therefore, if one contract developer is hired, the average cost assigns to a FTU process is approximate £700 to £$1,000$, provided 50% of the τ (80 to 120 hours in total) is spent on programing and debugging. Multiple developers could be hired to speed up the process development but the man-hour workload and cost per process development should be similar. dc is entirely a development relative control factor.

dc' - Average flexible-to-change process developing unit cost

A FTC process does not benefit from the system infrastructure. Its process developing costs more than a FTU process because of more workload and possibly extra resources required. It should be reasonable to assume that a FTC process development triple the cost of a FTU process, therefore the range of dc' is about £2, 100 to £3, 000. dc' is entirely a development relative control factor.

C_I - One-off initial cost

The initial cost is to establish the SCMS infrastructure, which includes hardware equipments, network facilities, purchasing of various software packages, and so on. Any customisation or bespoke development should not be accounted as part of the C_I. The initial cost range of £20, 000 to £30, 000 should be sufficient to establish a typical small SCMS, which involves 2 server computers, regular networking and data backup facilities, essential system software, database server software and developing environment together with a small supply chain management application (or some kind of middleware infrastructure). This cost range does not include a large-scale "solution pack" from dedicated supply chain management service providers as their price could vary to a large extent. By referencing to the cost model, the deviation of C_I should not affect the sensitivity analyses since this value adds on to the total cost C directly. Besides, the C_I is usually the most definitive variables in a real SCMS practice once the implementation scheme is decided. C_I should be classified as a control factor since it relies on the implementation decisions.

Table 4.1 summarises the whole value range set of the 13 variables discussed above to provide a quick reference for the commencing sensitivity analyses.

Table 4.1: Cost model variable value ranges

Variable	Type	Lower bound	Upper bound	Average value
T_f	Environmental	-	-	10,000 hours
F	Environmental	100	200	150
D	Environmental	20	30	25
τ	Control	80 hours	120 hours	100 hours
μ_m	Control	£1.15	£1.80	£1.48
μ_p	Control	£0.28	£0.34	£0.31
v	Environmental	3	6	4.5
w_1	Control	0.2	0.4	0.3
w_2	Control	0.45	0.7	0.575
x	Control	0.3	0.7	0.5
dc	Control	£700	£1,000	£850
dc'	Control	£2,100	£3,000	£2,550
C_I	Control	£20,000	£30,000	£25,000

4.3 Full Factorial DOE Analysis

When one looks into the details of an SCMS implementation, it should be noticeable that the defined 13 model variables are not independent to each other. For example, μ_m could be affected by D, and w_1 and w_2 are related to the v in certain ways. Some of the variable interactions exist in real world but not reflected in the cost model equations. It is hard to enumerate all the direct and intermediate interactions thoroughly or to describe the interactions in clear quantitative approaches. Most important, the interactions are probably dependent on each SCMS exercise. With these facts, the widely used Design Of Experiments methods are more appropriate than the SA method in the SCMS context.

The complete full factorial experiments for 13 factors with 3 value levels (i.e.

adverse, average and optimal values) require $3^{13} = 1,594,323$ tests, which is excessive to execute. A practical approach is to identify the important ones and perform the DOE experiments on the short listed variables only. The Plackett-Burman experiments (Plackett and Burman, 1946) are therefore applied prior to the DOE analysis to highlight the important variables with primary main effects. In the next DOE analysis, a 4 factors, 2 levels (average and optimal) full factorial experiment is performed on the important variables discovered by the Plackett-Burman experiments. In this circumstance only $2^4 = 16$ tests are required, which can be analysed by using the Analysis of Variance (ANOVA) table.

4.3.1 The Plackett-Burman experiments

The Plackett-Burman experiments design enables to identify the important variables with only a very limited number of experiments. It is a very efficient screening design when only main effects are of interest since the design does not permit one to distinguish between main effects and interactions. Variable interactions will be studied by means of the DOE approach afterwards.

For the temporal SCMS cost model, there are in effect 12 variables (excluding T_f as it is treat as a constant in the experiment setup of Table 4.1), there should be 16 runs to complete the Plackett-Burman experiments according to the design. As a mathematical cost model, it is not necessary to randomise the experiments to avoid residual errors which occur in many product making experiments. Three dummy variables ($DV1$, $DV2$ and $DV3$) are added to the experimental design (Table 4.3) to satisfy the condition that the experiment run N should be a multiple of 4 while variable count should be $N - 1$.

The design of 16 run Plackett-Burman experiments can be found in relevant research literature (Plackett and Burman, 1946; Cosier, 1987). Table 4.3 illustrates the details of the design and the results of 16 experiments. Each variable has 2 value levels (which are symbolised as + and -) to be fed into the cost model for outputs. The actual figures for the 2 levels are from Table 4.1, where "-" stands for the average value and "+" for the optimal value that reduces the total cost (it could be either the lower bound or the upper bound value depends on respective influence).

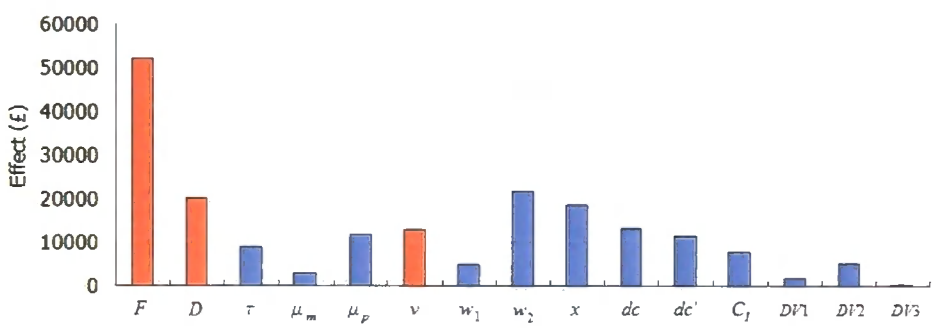


Figure 4.1: Variable effects of the Plackett-Burman experiments

The bottom line of Table 4.3 are effects of variables that derived from the formula of:

$$E_A = \frac{\sum R_{A-} - \sum R_{A+}}{8}$$

where E_A is the effect of variable A (symbol for any of the 12 model variables), $\sum R_{A-}$ is sum of results with A-, $\sum R_{A+}$ is sum of results with A+.

Table 4.2: Value list for the Plackett-Burman experiments

Model Variable	T_f	F	D	τ	μ_m	μ_p	v	w_1	w_2	x	dc	dc'	C_I
Average (-)	10,000	150	25	100	1.48	0.31	4.5	0.3	0.575	0.5	850	2,550	25,000
Optimal (+)	10,000	100	20	80	1.15	0.28	6	0.4	0.45	0.3	700	2,100	20,000

Table 4.3: The Plackett-Burman experiments

Run	Variables															Result (£)
	F	D	τ	μ_m	μ_p	v	w_1	w_2	x	dc	dc'	C_I	$DV1$	$DV2$	$DV3$	
1	+	+	+	+	-	+	-	+	+	-	-	+	-	-	-	183,167
2	+	+	+	-	+	-	+	+	-	-	+	-	-	-	+	197,130
3	+	+	-	+	-	+	+	-	-	+	-	-	-	+	+	216,251
4	+	-	+	-	+	+	-	-	+	-	-	-	+	+	+	216,845
5	-	+	-	+	+	-	-	+	-	-	-	+	+	+	+	256450
6	+	-	+	+	-	-	+	-	-	-	+	+	+	+	-	233,365
7	-	+	+	-	-	+	-	-	-	+	+	+	+	-	+	251,830
8	+	+	-	-	+	-	-	-	+	+	+	+	-	+	-	187,989
9	+	-	-	+	-	-	-	+	+	+	+	-	+	-	+	205,843
10	-	-	+	-	-	-	+	+	+	+	-	+	-	+	+	247,467
11	-	+	-	-	-	+	+	+	+	-	+	-	+	+	-	231,118
12	+	-	-	-	+	+	+	+	-	+	-	+	+	-	-	201,950
13	-	-	-	+	+	+	+	-	+	-	+	+	-	-	+	257,279
14	-	-	+	+	+	+	-	+	-	+	+	-	-	+	-	239,859
15	-	+	+	+	+	-	+	-	+	+	-	-	+	-	-	245,980
16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	328,322
Effect	*51,971*	*20,127*	8,695	3,057	11,735	*13,031*	4,971	21,860	18,684	13,313	11,502	7,731	1,760	5,270	332	

Figure 4.1 is the bar chart representation of the variable effects derived from the Plackett-Burman experiments (i.e., the bottom line of Table 4.3) to provide better readability. The effects of environmental factors have been highlighted in italic font and orange color bar to distinguish from the rest of control factor effects. Process type count F and process frequency D have remarkable effects on the model outputs but they are not attributes that SCMS implementation can change. Therefore, control factors of w_2, x, dc or dc', μ_p should be further investigated in the DOE analysis due to their greater effects on the total cost. The Lorenz Curve curvature v may also be reviewed as a result of its high effect and intricate influences.

4.3.2 The DOE ANOVA table

With the 4 important main control factors (w_2, x, dc', μ_p) identified by means of the Plackett-Burman experiments, it is ready to continue the full factorial DOE analysis, so that both main effects and interactions can be revealed. A simplified ANOVA table is constructed since there is no need to repeat and randomise the experiments to overcome residual errors or deviations.

Similar to Table 4.2 for the Plackett-Burman experiments, a 2-level (average and optimal) value matrix is generated (Table 4.4) to be fed into the cost model equations. Factor codes (i.e., A, B, C and D) are assigned underneath the 4 designated variables for a neat ANONA table presentation. Those without factor code are treated as invariables in the experiments, therefore their optimal values are left blank.

Table 4.4: 4-factor full factorial DOE analysis variable values

Model Variable	T_f	F	D	τ	μ_m	μ_p	v	w_1	w_2	x	dc	dc'	C_I
Factor Code						B			C	A		D	
Average (-)	10,000	150	25	100	1.48	0.31	4.5	0.3	0.575	0.5	850	2,550	25,000
Optimal (+)						0.28	6		0.45	0.3		2,100	

Table 4.5: 4-factor full factorial ANOVA table

Cell Group	Factors				2 Factor Interactions						3 Factor Interactions				4 Factor Interaction	Output (£)
	A	B	C	D	AB	AC	AD	BC	BD	CD	ABC	ABD	ACD	BCD	ABCD	
1	-	-	-	-	+	+	+	+	+	+	-	-	-	-	+	328,322
2	-	-	-	+	+	+	-	+	-	-	-	+	+	+	-	308,916
3	-	-	+	-	+	-	+	-	+	-	+	-	+	+	-	297,470
4	-	-	+	+	+	-	-	-	-	+	+	+	-	-	+	282,283
5	-	+	-	-	-	+	+	-	-	+	+	+	-	+	-	319,492
6	-	+	-	+	-	+	-	-	+	-	+	-	+	-	+	300,085
7	-	+	+	-	-	-	+	+	-	-	-	+	+	-	+	288,666
8	-	+	+	+	-	-	-	+	+	+	-	-	-	+	-	273,478
9	+	-	-	-	-	-	-	+	+	+	+	+	+	-	-	298,997
10	+	-	-	+	-	-	+	+	-	-	+	-	-	+	+	287,353
11	+	-	+	-	-	+	-	-	+	-	-	+	-	+	+	274,520
12	+	-	+	+	-	+	+	-	-	+	-	-	+	-	-	265,408
13	+	+	-	-	+	-	-	-	-	+	-	-	+	+	+	290,167
14	+	+	-	+	+	-	+	-	+	-	-	+	-	-	-	278,523
15	+	+	+	-	+	+	-	+	-	-	+	-	-	-	-	265,716
16	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	**256,603**
Main & Interaction Contribution	22,678	8,817	25,963	13,838	0	-2,766	-3,459	-13	0	-1,688	0	0	422	0	0	

4.3.3 Interaction analysis

Figure 4.2 is the bar chart representation of the main and interaction contributions appearing at the bottom row of the ANOVA table (Table 4.5). It is obvious that the 4 factors have distinctive main effects than their interactions, if exist. The w_2 weights most due to high developing costs and x, dc' and μ_p are also very important in achieving cost-effectiveness.

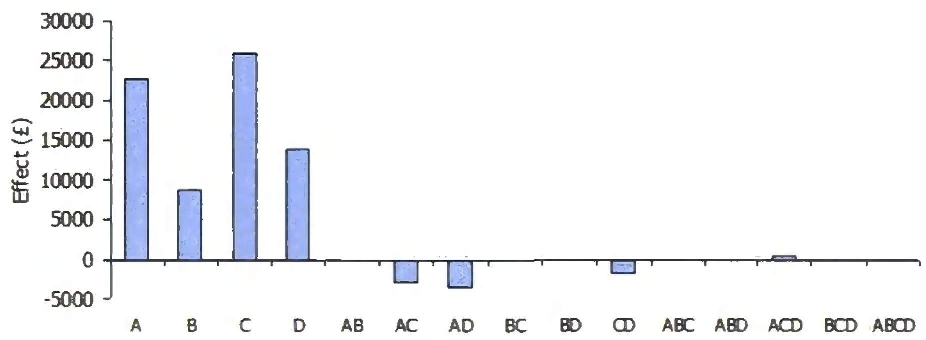


Figure 4.2: Main and interaction contributions

The weak interactions among the 4 variables are actually beneficial to the SCMS exercises since variables don't influence each other notably when adjusting each variable to achieve an optimal total cost. To support this conclusion, the non-zero 2-factor interactions, which are AC, AD, BC and CD interactions according to Figure 4.2 or Table 4.5, are plotted in Figure 4.3 respectively. In this diagram, the blue and magenta lines represent the high and low cost effect respectively. If the lines are parallel then there is no interaction between the 2 factors. The length of green lines indicates the degree of interactions which have been calculated in the ANOVA table already. It proves again that the 4 factors are nearly independent to each other according to their proximate parallel effect lines. As all the high factorial interactions are either zero or tiny, it can be asserted accordingly that the 4 factors do not influence each other so they can be adjusted independently towards the optimal cost output without worrying about their interactions.

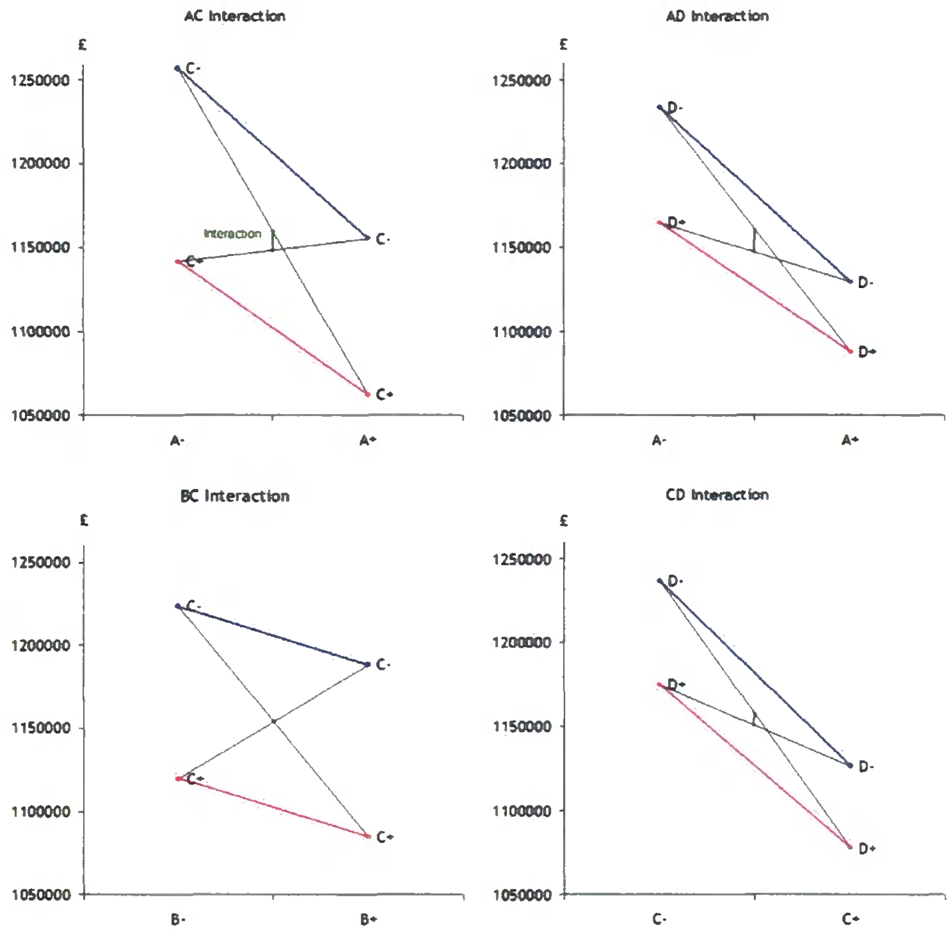


Figure 4.3: 2-factor interactions plotting

4.3.4 Full factorial DOE conclusion

According to the supply chain environment and optimal values for the 4 factors, the best cost-effective setup is achieved at the condition when A+, B+, C+ and D+, as a result of the weak interactions among the factors. This full factorial experiment demonstrates a practical method to identify the important SCMS model variables and their interactions, so that SCMS planners can achieve the cost optimisation with limited information and conditions. The environmental factors, although have significant effects on the total cost, can not be optimised at the SCMS implementation level. Therefore, the control factors are what to be paid more attention to.

The fundamental defect of the DOE method in the SCMS context lies in the limited amount of variables and their value levels that can be involved in one experiment. The negligible 2 value level is too limited to find out the best practice. In addition, the average and optimal values are open to subjective interpretation or bias, which adversely affects the accuracy and validity of the analysis. To overcome the limitations of DOE, a different experimental method is introduced in the following sections to study how model variables contribute to the final cost respectively at continuous value series.

4.4 Individual Model Variable Analysis

In sensitivity analysis it is a common practice to investigate what effect a model variable produces on the output by changing One factor At a Time (OAT) (Daniel, 1973), whilst keep all other factors fixed to their average values. The OAT method, although logically sounds, has received many criticisms due to its non-explorative defect(Czitrom, 1999). This is to say, when one factor is being inspected with other factors fixed to their average values, the full condition space can never be explored. Due to this limitation, it is not suitable to study the factor interactions with the OAT method. And for the same reason, it is helpless in finding the best setting for a cost model practice.

Despite these weaknesses, the OAT method is still an effective method to study the approximate (since interactions cannot be taken into account) individual variable sensitivity at fine grain, compared to the design of experiments. The weak factor interactions of this cost model, discovered by the DOE, also endorses the choice of the OAT. In this section, model variables related to supply chain flexibility, process variability and difficulty will be analysed with the OAT approach to investigate their individual cost contribution, so that SCMS planners can establish an overview of the model variable characteristics. With the uniqueness of each supply chain practice in mind, it is impossible to explore the full list of possible scenarios to make up a quick reference manual. This section is a demonstration of using appropriate methods to achieve cost effectiveness from the system construction perspective.

At the early stage of supply chain management system construction, a crucial step is to determine the appropriate system infrastructure to constitute the skeleton of the whole SCMS. The sense of "appropriate" in this context means it should be able to:

1. Reduce the initial cost C_I (price of purchasing, training cost, initial setup cost, etc.)
2. Have long system life time (by applying mainstream technologies that will be continuously supported by the software industry)
3. Develop as many known process types as possible before system functioning (higher w_1)
4. Reduce developing cycle (lower τ) and process developing cost (lower dc)
5. Enable easy and cheap expansion of existing infrastructure (lower dc')

Point 1 and 2 rely on the software market condition at the time, therefore they have little to do with this cost model. However, it does require a broad vision of the state-of-the-art information technologies to make the right choice. Obsolete or unpopular technologies can be very expensive to maintain, since the knowledge or specialists are difficult to find. In the worst cases, large proportion of conjunctive function modules may have to be abandoned altogether to just accommodate a small change. Supply chain system users usually lack sufficient skills to continue the development on their own in this circumstance. Therefore it is suggested to keep on the mainstream technologies to minimise risks, if possible.

Point 3 relies on the feature set of the selected software product. So far, hundreds of management information systems have been brought to the market. Many of them were targeted to a specific customer segment. In the past, when choices were limited, companies bought generic SCMSs and tailored them if possible to meet their own requirements. From time to time, software products continued emerging to grasp the gradual fine-grained market. Therefore, there should exist products, for instance SCMS for the manufacturing industry, SCMS for the automotive manufacturing industry, and next level down, SCMS for the European automotive industry and so on. In this example, if the company happens to belong to the European automotive industry, it will be most appropriate to adopt the specific SCMS product as in principle it should provide the broadest coverage of functions needed without extra adjustment or re-development. Benefits of applying appropriate SCMS product can also come from easy data communication with other supply chain partners if they utilise the same system. On the other hand, with the same example, if the company is multi-national, the European version may cause difficulties in aligning with local conventions and trading legislation for the non-European subsidiaries. In this scenario, the more generic "SCMS for automotive manufacturing industry" version might be a better choice overall. Therefore, it depends on an in-depth review of available functions against demands to determine the most appropriate system.

Point 4 and 5 indicate a better modular and open system structure that is cheap with regard to functional extensions. Mainstream technologies are once again recommended because the knowledge can be easily acquired from the labour market.

4.4.1 Cost contribution of the process uncertainty x

In terms of process uncertainty, the system infrastructure ascertains how many process types fall in the flexible-to-use category, namely they can be supported as easy functional extensions at low developing costs. In the model variable list (Table 3.1), only argument x (process type uncertainty share) is directly related. By inspecting the equations of cost elements (i.e., $C_I, C_M, \cdots, C_X$), it is clear that the only relevant cost element C_D is a first order function to x. Therefore the final cost C has a linear relationship with x as well.

To evaluate the cost contribution of x against the total cost C, an experiment is performed based on the supply chain environment established earlier (Table 4.1). The used values are shortlisted in Table 4.6. The range of x is extended to the full scope to draw a complete picture of its cost effect.

Table 4.6: Value list of process uncertainty x OAT analysis

Variable	T_f	F	D	τ	μ_m	μ_p	v
Value	10,000	150	25	100	1.48	0.31	4.5
Variable	w_1	w_2	dc	dc'	C_I	x	
Value	0.3	0.575	850	2,550	25,000	$[0, 0.1, \cdots, 1]$	

By substituting these values into the cost model, a result set can be obtained as Table 4.7 and visualised in Figure 4.4. According to the results and the diagram, it is clear that process uncertainty x has linear relation to the total cost C. It is guaranteed that a higher uncertainty rate always results in a more expensive overall cost, unless a certain process occurrence rate is too low that results in cheaper manual handling costs (which means too many processes are developed than necessary. In terms of this cost model, it indicates excessive w_2 is applied).

Table 4.7: Process uncertainty x OAT analysis

The fixed cost elements in this experiment:

Cost Element	C_I	C_M	C_S	C_R	C_X	Sub Total
Value	25,000	3,138	19,446	76,843	19,020	143,447

x	C_D	C (£)
0	111,563	255,010
0.1	126,225	269,672
0.2	140,888	284,335
0.3	155,550	298,997
0.4	170,213	313,660
0.5	184,875	328,322
0.6	199,538	342,985
0.7	214,200	357,647
0.8	228,863	372,310
0.9	243,525	386,972
1	258,188	401,635

In the model analysis section of (Gebauer and Schober, 2006), process uncertainty impact on the total cost was inspected at coarse grain of $p = 0.8$, $p = 0.5$ and $p = 0.2$ (p was the uncertainty mark used in that research). Despite the fact that the cost figures in these two models are not directly comparable, the cost transition trends should be consistent under similar conditions. However, Gebauer and Schober's model showed a unexpected turning point in the mid-

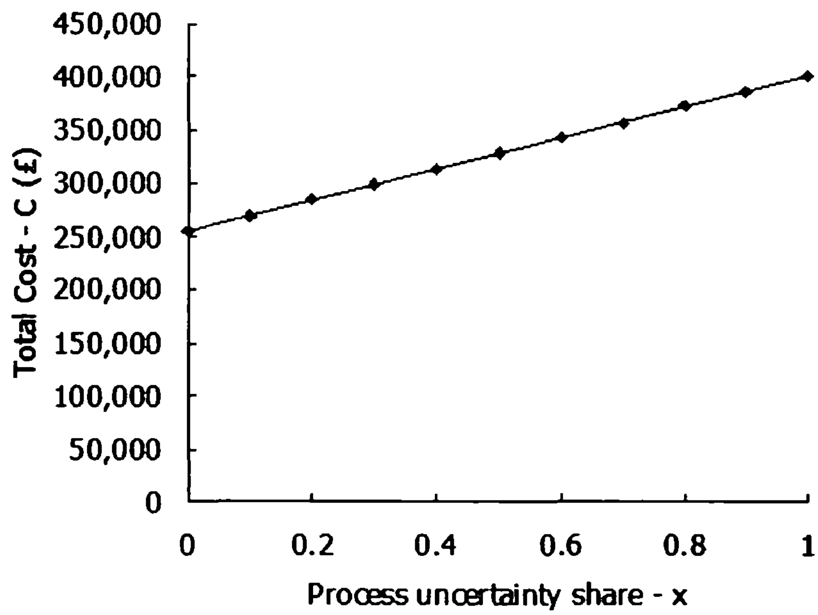


Figure 4.4: Process uncertainty share x OAT analysis line chart

dle, for instance $C_{p=0.8} < C_{p=0.2} < C_{p=0.5}$ which was listed in Table 1, 2 and 3 of that paper. This happened at medium process variability level such as when $v = 0.4$ (Figure 4, Page 17). This result might be the inaccuracy sourced from the timeless characteristic of the Gebauer and Schober's cost model.

4.4.2 Cost contribution of the process variability v

The process variability v (i.e. the Lorenz Curve curvature) is the variable incorporated in most cost elements of this cost model. It is known to have important cost effect through the Plackett-Burman experiments. Now it is worth taking a closer look at how it influences the total cost. With the same experimental environment, an OAT analysis is established to observe the model outputs over a series of v values in a stacked bar chart so that transition of both the total cost and each cost element can be read easily. The variability v

is increased linearly within the range of $v \in [0.01, 20]$. All the other variables are retained constant as shown in the Table 4.8. Table 4.9 lists the outputs for all cost elements along with the increase of v. The last column is the total cost C that sums up the associated cost elements.

Table 4.8: Value list for process variability v OAT analysis

Variable	T_f	F	D	τ	μ_m	μ_p	C_I
Value	10,000	150	25	100	1.48	0.31	25,000
Variable	w_1	w_2	x	dc	dc'		v
Value	0.3	0.575	0.5	850	2,550		$[0.01, 2, 4, \cdots, 20]$

Table 4.9: Process variability v OAT analysis

v	C_I	C_M	C_S	C_D	C_R	C_X	C (£)
0.01	25,000	46,048	54,169	184,875	67,855	73,238	451,184
2	25,000	16,448	37,060	184,875	74,055	45,085	382,523
4	25,000	4,478	22,332	184,875	76,562	22,925	336,172
6	25,000	1,027	12,650	184,875	77,285	10,642	311,479
8	25,000	213	7,007	184,875	77,455	4,829	299,380
10	25,000	42	3,855	184,875	77,491	2,211	293,474
12	25,000	8	2,117	184,875	77,498	1,032	290,530
14	25,000	1	1,162	184,875	77,500	491	289,029
16	25,000	0	638	184,875	77,500	238	288,251
18	25,000	0	350	184,875	77,500	117	287,842
20	25,000	0	192	184,875	77,500	58	287,625

Figure 4.5 plots the value matrix of Table 4.9 in a stacked bar chart style. As it reveals, the process variability v has obvious influences on C_X, C_S and C_M. The final cost C presents a downward converging trend towards $(C_I + C_D + C_R)$ along with the increase of v. At the extreme inequality side of the trend (where

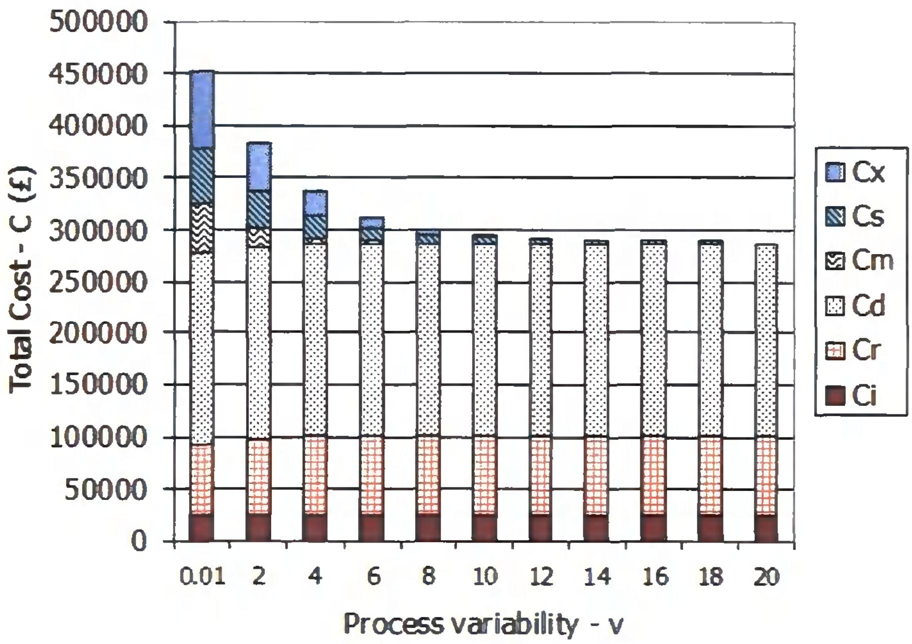


Figure 4.5: Process variability v OAT analysis bar chart

$v = 20$), process occurrences concentrate on a single process type which was developed in the very beginning before system functioning (i.e., within the w_1 proportion). The total cost drops 50% when the process variability v reaches this extreme inequality level, because at this condition all process occurrences are of one process type handled by a software routine which is developed before system functioning.

4.5 Summary of Sensitivity Analysis

In this chapter, the DOE and Plackett-Burman based sensitivity analyses and OAT method are introduced in order to identify the important model variables of a supply chain exercise as well as cost contributions of some individual variables. Although not all the model variables are covered, this analysis can be extended to larger scale by means of statistical analytic tools such as Minitab[2].

These types of analyses are considered basic as they are performed directly at the model variable and equation level. For the purpose of providing a practical tool to achieve SCMS cost effectiveness, it is favorable to tackle the problems from the system implementation perspective. In the next chapter, with respect to modern information technologies, some more pragmatic information system related topics will be addressed to guide SCMS implementations by means of the temporal cost estimation model.

[2]A statistical software targets to industrial quality improvement. see website http://www.minitab.com for details.

Chapter 5

Evaluation of SCMS Implementation Strategies

In this chapter the SCMS cost effectiveness objective is reviewed from the implementation perspective by converting system implementation decisions into set of appropriate values of cost model variables to discover the optimal cost by means of this cost model.

5.1 SCMS Implementation Schemes

The fundamental challenge when constructing a SCMS is to determine the suitable information system platform and technologies, so that the cost effectiveness can be achieved whilst business demands are satisfied. The introduced cost model is not a tool to actively initiate a SCMS "all-purpose best practice", however it is capable of discovering the best in a short list of candidates.

Every popular SCMS implementation approach comes with its strengths and weaknesses, and these conditions change all the time along with the evolution of technologies. To date, very few standards exist for this type of software as a whole (Hannus et al., 2003). The best choice would be a trade-off of business type, existing software system structure and technologies, available resources such as labour and skills, business strategies and developing plan, and so on, at that time. Suppose the supply chain management *system* (that comprises of both information facilities and system users) is not a premise for commercial activities (which assumes the company can handle any kind of task manually), then the best choice will be the cheapest overall.

Based on the state-of-the-art information technologies, SCMS end users can obtain their systems mainly in four typical schemes:

1. COTS

   COTS stands for "Commercial Off The Shelf" system. It is a ready made software package that is available from the market. Users can make use of it with little or no modification.

2. Framework based

   Framework based product is software package that provides lower level software assemblies to be organised together to comprise a workable SCMS. As many common functions have been well developed in the frameworks, it is usually less challenging to create a SCMS based on them than starting a new system design from beginning.

3. Proprietary development

   There are basically two types of proprietary development: the company either does the software development from scratch on its own (in-house development), or by sub-contracting to software vendors as a project (turnkey system).

4. Open source based

   The company utilises open source products to constitute its information system. Due to the boom of the open source campaign, this scheme has drawn more and more industrial attentions. It is unique regarding system cost and re-development compared to regular implementation schemes therefore listed separately.

In the following sections, these four schemes are discussed by enumerating their strengths and weaknesses with respect to the supply chain characteristics. In this research, the attentions have been paid to the cost and productivity aspects from the SCMS planners' view without getting into excessive technical details, which in the end will also reflect on the cost as a commercial application. Considering the target cost estimate model users, this cost-driven comparison should be appropriate for the "best practice" purpose.

5.2 Scheme Comparison

5.2.1 COTS

COTS, short for Commercial Off-The-Shelf, is the term for close-sourced software products that are ready made and available to the public from the software market. COTS software is usually well tested and maintained by software vendors at high price, and may offer significant saving if it fits the user's supply chain environment. Details about its strengths and weaknesses can be listed as below:

- Strengths
  - Well designed, generally high quality as COTS is usually verified in many ways by the vendor, public testers and customers.
  - Rich source of knowledge due to its large user base. Tasks like user training, trouble shooting and maintenance are well-documented and less challenging.
  - Information exchange with business partners who implement the same system is likely efficient and reliable.
- Weaknesses
  - The majority of COTS functions provided are usually so called "industrial standard processes", but not "customer's specific processes". It is quite common that COTS users abandon a lot of native functions because of the inconvenience caused by the inconsistency, if system flexibility is insufficient. These gaps will be magnified in chained processes and deemed "non-conformity" eventually.
  - COTS suppliers lack in-depth understanding of customer's specific demands and the associated trading environment. The price for modifications or functional extensions is usually expensive to the users, so only essential modifications will be requested in most case. For this reason, the quality and range of system customisation / modification is very limited.
  - COTS suppliers tend to maintain the "main trunk" of their product which contains only the intersection of common functions presented to the product users. The customisations and modifications for individuals will probably not be included in the next version. It

is challenging for those modified implementations to follow up the upgrade, especially when the cost-effectiveness has to be taken into account.

- Flexibility of COTS is usually low since business logic is deeply embedded inside with little space left for end users' extension.
- Close sourced. COTS users are forced to consult the supplier for modifications or implementation details. To obtain this kind of support, COTS users are tied to maintenance contracts, at a cost.

- Existing products

  There are quite a few SCMS products on the software market despite the list changes all the time. The market research statistics by AMR Research[1] and ARC Advisory Group[2] indicated that the worldwide SCMS market would reach or exceed $8 billion by year 2010, the leading providers for the moment were SAP, Oracle and Infor. However, selecting a suitable SCMS products has never been as straightforward as choosing machines or general purpose retail software applications that have comparable specifications and features against list prices - the price of a supply chain COTS system is always subject to individual implementations. As a result, supply chain COTS system surveys (e.g., the research by the OR/MS Today magazine[3] in year 2003) have attempted to structure the selection process by lengthy questionnaires to help narrowing down the choice list and respective price ranges.

[1] AMR Research was founded in 1996, Boston. It provides comprehensive research and advisory services for supply chain and IT executives. It aims to help organisations to identify opportunities and improve operational procedures to get the most out of IT investments.

[2] Founded in 1986, ARC Advisory Group is a research and advisory firm for manufacturing, energy, and supply chain solutions.

[3] A magazine for members of the Institute for Operations Research and the Management Sciences (INFORMS). Published by the Lionheart Publishing, Inc. URL: http://www.lionhrtpub.com

The Technology Evaluation Centre (TEC) has an interactive service to compile a COTS comparison report from implementation details supplied to its questionnaire. In the generated report, a shortlist of suitable COTSs are recommended which comply with the functional requirements as well as other aspects such as budget constraint. By feeding into the questionnaire with the same supply chain conditions described in Table 4.1 and the assumption of a supply chain in the manufacturing business section, a fraction of the recommended COTS list from the TEC report is copied to Table 5.1. This product shortlist and price range are for reference purpose only due to its time dependency, not to mention the functional demands for each SCMS implementation spread on a very wide spectrum.

Table 5.1: COTS product examples

Vendor	Product	Product Description	Price Range
SAP	SAP Supply Chain Management	SAP SCM is part of the SAP Business Suite. It enables collaboration, planning, execution, and coordination in the entire supply network to adapt the supply chain processes to an ever-changing competitive environment.	£30,000 - £150,000+
i2	i2 Supply Chain Management	i2 Supply Chain Management Solutions are geared towards solving customer-specific business objectives. They are built upon industry best practices and leverages years of experience from large quantity of implementations. i2 solutions integrate with data, processes, and third party systems.	£35,000 - £100,000+
Oracle	JD Edwards EnterpriseOne Supply Management	Oracle JD Edwards EnterpriseOne (formerly PeopleSoft) Supply Management comprises fully-integrated planning/forecasting and fulfillment applications. The supply chain products deliver enhanced productivity and efficiency for product-driven industries. In addition, the integration between PeopleSoft Advanced Planning and fulfillment software enables instantaneous response to events that impact a company's supply chain.	£25,000 - £150,000+
Infor	Infor SCM	Infor SCM meets the challenge with specialised functionality that takes into account the different supply chain perspectives and unique business challenges of manufacturers, retailers, and transportation and logistics service providers.	£20,000- £100,000+
...	...	... ...	...

Conventionally, 10% of the purchasing price can be used as annual maintenance cost for rough estimation. This cost will be added on to the μ_p.

Most successful commercial supply chain management systems have modular structure to satisfy the demands of various business sections. This requires a good degree of core module flexibility and complexity which, on the other hand, could impact the software price retroactively. The actual price for a COTS implementation will be much higher than the minimal prices listed when costs for customisation work and implementation consulting are counted in. The steep cost will be hard for low budget supply chain projects to embrace, especially when only a small function subset of the COTS is required.

5.2.2 Framework

The term framework in this context stands for a set of underpinning software objects and function assemblies that are common to most supply chain practices. Customer SCMS can be developed on top of these low level functions and objects at low cost and short time scale. Sometimes the framework is also regarded as "business objects layer" or "middleware" in the multi-tier software development area. Examples of the framework are workflow engines, data exchange interfaces that can exchange business information in the common format like Electronic Data Interchange (EDI), eXtensible Markup Language (XML) via public network. Framework based development is easier than developing from scratch but a certain level of software design and developing skills are still essential. The strengths and weaknesses are enumerated as follows:

- Strengths
  - Frameworks are highly flexible to be used for any type of SCMS implementation.

- Compared to COTSs, framework products usually cost much less to buy.
- Compared to proprietary software, the framework based development is less difficult because the developers can focus on high level building blocks only.
- Once developers obtain sufficient experience with the chosen framework, the overall developing productivity can be maintained at a high level.

• Weaknesses

- There are very few functions ready for use in the beginning.
- Software developing skills are still essential.

• Existing products

So far, dedicated supply chain application frameworks still do not exist. However, a supply chain management system, despite its unique environmental uncertainty, still falls in the functional scopes of data exchange, application integration and inter-connectivity, inside out the supply chain participator. These functionalities are not entirely new to the software industry. Existing Enterprise Application Integration (EAI) frameworks, can be applied to build the SCMSs. Table 5.2 lists some popular frameworks.

Table 5.2: Framework product examples

Vendor	Product	Product Description	Price Range
Microsoft	BizTalk	BizTalk server is Microsoft's integration and connectivity server solution. It provides a solution that allows organizations to more easily connect disparate systems. In addition to integration functionality, BizTalk also provides strong durable messaging, a rules engine, EDI connectivity, Business Activity Monitoring (BAM), RFID capabilities and IBM Host/Mainframe connectivity.	£6500 for standard, £27000+ for enterprise
Oracle	BEA WebLogic	WebLogic is a product family that delivers multi-tier application servers and integration that unifies all the components of business integration (process management, data transformation, trading partner integration, and user interaction) in a flexible, easy-to-use environment.	£20000 per CPU
...	...	... ...	...

Generally speaking, the framework based scheme best suits companies with software developing capability whilst aiming to retain a high level of flexibility and control. It should also be considered as a good choice under circumstances when: joining immature supply chains (where changes happen more frequently); in a small scale supply chain (where COTSs are expensive and over bloated for the functions demanded); participating in multiple supply chains (where a suitable COTS is difficult to find); or intending to integrate internal production systems seamlessly (as it is difficult to firmly integrate with COTSs without sufficient implementation details).

5.2.3 Proprietary development

The proprietary scheme indicates the supply chain company does the SCMS development from scratch by itself (called *in-house* system) or by sub-contracting to a dedicated software supplier (called *turnkey* or *bespoke* system). In this way, the company retains maximal control and flexibility to achieve exactly what is required. The disadvantages are also distinct due to its high development risks and barrier of technical skills. The details of its respective strengths and weaknesses are:

- Strengths
  - It has maximal flexibility as the developing team is free of restrictions to achieve the targets.
  - By processing and presenting information in local terms and styles, end users can easily adapt to it. In this way, the best system performance and company culture compliance can be achieved, which results in cost-effectiveness.
  - It is able to deliver functions that fit the exact requirements.
- Weaknesses
  - Error-prone. Proprietary system is hardly as well tested as COTSs due to the high cost and small user group.
  - As a result of supply chain uncertainty, there is a huge risk that the SCMS project can be out of control attributing to the side-effect of developing flexibility.

- The system architecture tends to decay after excessive changes under current function-oriented software techniques. The final SCMS can be full of disordered code and bugs. A low quality SCMS causes cost penalty as a result of system breakdown and data lose.
- The proprietary SCMS will be more and more difficult to maintain along with the evolution of supply chain.

Considering the huge risks and technical barrier to follow this scheme, the proprietary developing scheme should be considered as an extreme case where supply chain functions are very simple and clearly defined or when the developing team is highly experienced. In some cases, proprietary development can be adopted as complementary to COTS or framework based SCMS, to eliminate performance bottlenecks or functional gaps.

5.2.4 Open source approach

Over recent years, the Open Source Software (OSS) has received many public attentions for its dramatic impacts on the software industry. An OSS based SCMS can provide a ready-to-use software package together with the source code for further modification, all for free.

It sounds like a perfect choice in the first place, but why are companies still spending so much money on COTSs? Leaving politics and preferences aside, there are still a few issues that hinder its popularity in the enterprise supply chain realm:

- First of all, due to limited budget and the volunteer-based developing group, open source projects struggle to compete with commercial SCMSs with regard to quality and reliability due to the expensive cost of thorough system test.

- For the same reason above, documents and technical supports are likely to be insufficient for end users to carry out implementations and extensions.

- Further, the objectives of open source projects are normally of voluntary developer's interests or for research purposes. End users can hardly steer the direction as there is no firm commitment between the original developing team and end users, unless particular agreements are settled.

- OSS projects might be discontinued which will leave users alone owing to lack of contractual constraints between developers and users. This possibility is a huge risk to end users from business continuity point of view.

- Lastly, commercial users might be obliged to publish the source code of their modifications to satisfy copyright constraints. This might put the company confidential information at risk from potential malicious attacks as software security flaws commonly exist. It is especially the case for SCMSs as a result of their Internet dependency.

- Existing products

  In recent years the open source campaign has gradually stretch into the enterprise application area such as ERP or financial applications. Supply chain management systems are surely one of them despite the common incompleteness due to lack of resources and volunteers. There is still a long way to go before a full-fledged open source SCMS emerges. Table 5.3 presents examples of current open source SCMSs or applications cover supply chain management functionality.

Table 5.3: Open source product examples

Vendor	Product	Product Description	Price Range
Compiere	Compiere ERP and CRM	Compiere is a full featured open source ERP system with extensions of CRM, shop floor control, supply chain management, quality management, modbus interface, barcode extension, etc.	-
-	ADempiere ERP Business Suite	ADempiere Business Suite is a community fork of Compiere. It focuses on the community that includes subject matter specialists, implementers and end-users.	-
-	SigmaSCM	SigmaSCM is an completely web-based open source supply chain management system that consists of components including manufacturing management,warehouse management order fulfillment and B2B system integration.	-
	...	... ...	...

In general, the open source scheme should be considered by balancing company information policies with financial situation comprehensively. It is suggested that there should be a relatively strong developing team to take the open source route due to the common incompleteness and defects in open source projects. A close cooperation with the open source project team might be a good choice if applicable. In addition, the open source approach may require special non-financial considerations such as risks of commercial confidentiality exposure, which is outside the scope of this study.

5.3 Quantitative Analysis

Having introduced the SCMS implementation schemes, it is feasible to look into the details of how much each scheme will cost by referring to the cost model. In the model sensitivity analyses section, the 13 model variables have already been split into two categories: environmental factors and control factors (Table 4.1). For a fair implementation scheme comparison, the environmental factors should be kept identical. While for control factors, the cheap but reasonable choices based on the supply chain environment are specified for the four schemes. Under this premise, the respective cost lines can be drawn along with the progress of process development. This is the plan of the scheme comparison experiments.

5.3.1 Case study of scheme comparison

The following scheme comparison experiments are still based on the small supply chain scenario specified in Table 4.1. The control factor values for the four implementation schemes are determined respectively by the price information collected and by experience or assumptions due to their implementation dependency. It is arguable that more or less personal bias in specifying the values is involved, but the model results are subject to this specific case study only.

Apart from the common supply chain environment conditions, it is supposed the COTS has 40% initial process coverage before any customisation ($w_1 = 0.4$, which is fairly high) and a open source product is available to have initial process coverage of 20% ($w_1 = 0.2$) as the opponent. The COTS in this example is considered well constructed (which is to say it has less bugs with well-designed maintenance routines) hence it introduces a lower unit running cost ($\mu_p = 0.18$) than the others. Due to the close-sourced nature, the COTS is probably slower ($\tau = 120$) and more costly ($dc = 1000$, $dc' = 3000$) to have functional extensions. Variables for other schemes are specified accordingly by referring to their respective strengths and weaknesses as discussed previously. Table 5.4 collects the allocated values for the four implementation schemes.

Table 5.4: Value matrix for implementation scheme comparison

Environmental and static variable values:

Variable	T_f	F	D	μ_m	v
Value	10,000	150	25	1.48	4.5

Control factor values:

Variable	COTS	Framework	Proprietary	Open Source
$C_I{}^*$	26,000	12,500	6,000	6,000
μ_p	0.28	0.31	0.31	0.34
w_1	0.4	0	0	0.2
x	0.8	0.3	1	0.8
dc	1,000	700	-	850
dc'	3,000	2,100	1,500	2,550
τ	120	80	100	100

* Basic system cost (hardware and database, etc.) is set to £5,000 for a small SCMS with £1,000 added on for developing tool, plus application system cost

With the implementation scheme conditions specified, SCMS total costs can be calculated in pace with the developing progress w_2 to shape a cost trend line for each implementation scheme. Table 5.5 presents the cost result matrix at 10% increment interval of w_2. The cost transition can be clearly discovered by filling the cost figures into a curve chart (Figure 5.1). In this way, the cost results are directly comparable at same post-development levels. Considering the objective of cost effectiveness, a "fully automated SCMS" is neither the best choice for end users nor the intention of this case study. In effect, manual handling is more preferable if the overall cost is lower. Based on this perception, the overall minimal cost among the four schemes can be easily discovered together with the condition to achieve it. This result indicates the optimal implementation plan one should follow.

Table 5.5: Scheme costs comparison (Minimal values highlighted)

w_2	COTS (£)	Framework (£)	Proprietary (£)	Open Source (£)
0%	**213,691***	460,000	453,500	264,464
10%	236,142	375,997	376,792	**258,313**
20%	266,676	336,723	346,286	270,465
30%	301,607	**322,998**	**341,524**	292,816
40%	338,861	323,669	350,810	320,731
50%	377,288	332,311	367,541	351,596
60%	416,267	345,265	388,058	383,960
70%	-	360,476	410,373	417,037
80%	-	376,815	433,445	450,413
90%	-	393,672	456,728	-
100%	-	410,733	480,041	-

* The overall minimal cost.

The results reveal that for this experiment setting, the COTS is the winner. It reaches the overall optimal cost with no customisation at all. The common possible reason is either due to over-expensive developing cost (dc, dc', τ) or low total process frequency (D) that prevents computer automation from being the more economic choice. There could be more possibilities subject to respective implementation environment. For instance, a higher labour cost μ_m would encourage the degree of computer automation.

Surprisingly, the open source scheme demonstrates its strong competitiveness at the low w_2 band, despite its much lower initial process coverage (where $w_1 = 0.2$) than the COTS (where $w_1 = 0.4$) . The framework scheme has a

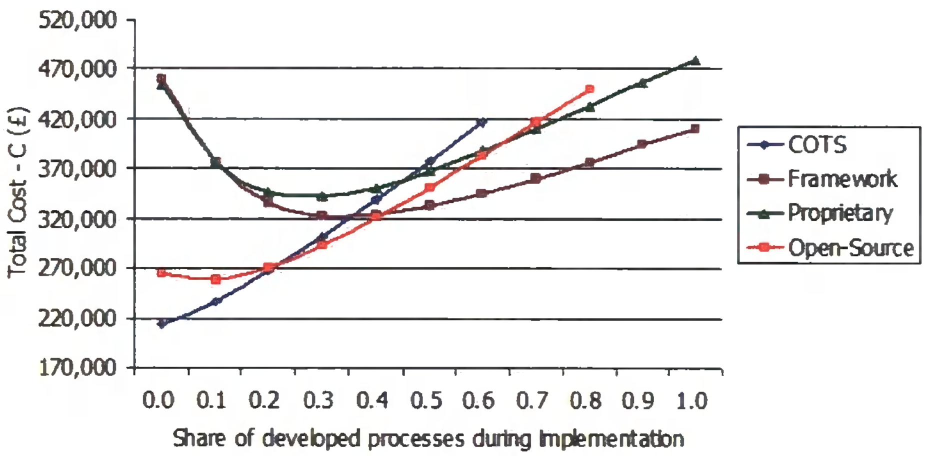


Figure 5.1: Curve chart representation of scheme costs comparison

U-shape cost trend line. The cost turning point is an important index as it is uneconomical to continue the process development after that point since the same total cost can be achieved by doing less system development. Nevertheless, at high w_2 band, the framework scheme shows its best competitiveness. The proprietary scheme loses its point at both low and high w_2 band therefore it is not recommended to take this approach unless the supply chain is really simple or small scaled.

The shape of a cost trend line is the combined effects of model variables and their interactions. There are many ways to influence the cost shape by justifying variable values. The relationship is in fact embedded in the model equations which are more descriptive than literal guidelines. To achieve an optimal SCMS implementation, this scheme comparison should be exercised on each SCMS practice.

5.3.2 Effect of supply chain size on the four schemes

So far, all the experiments were performed on a small supply chain experimental environment with process type count of $F = 150$, dramatic discrimination might exist in supply chains of different scales. To have a full picture of the cost trend against supply chain size F and process type share w_2, it can be achieved by considering the F and w_2 as independent variables to observe the cost transition in a 3-dimensional space. Figure 5.2 plots the cost results for each scheme as a surface in the range of $F \in [0, 500]$, which covers small to large size supply chains. The code to generate this diagram is attached in the appendix.

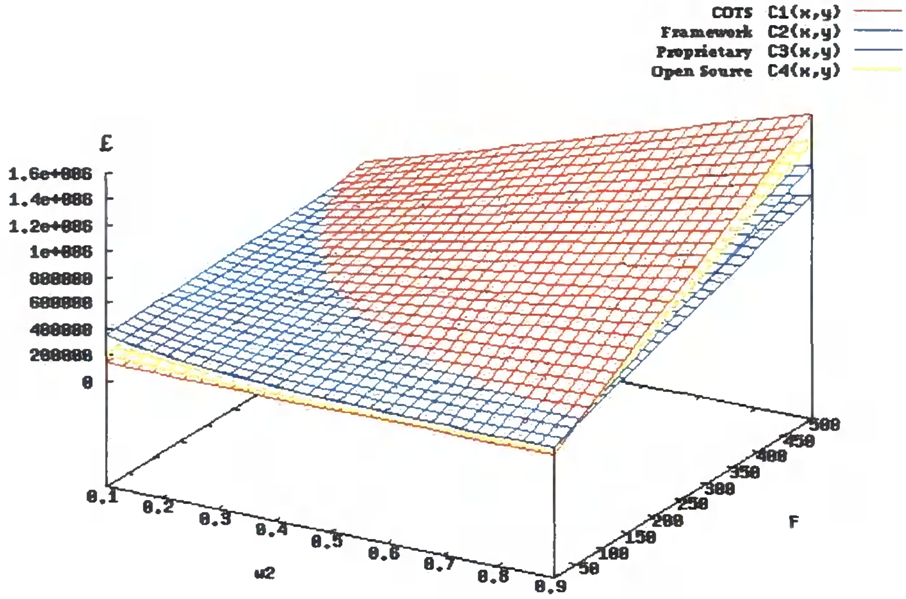


Figure 5.2: 3D plot of cost trend with variable F and w_2

According to the chart, the COTS scheme loses its competitiveness in large scale complex SCMS. It also shows that the framework scheme is a good choice particularly for large size supply chain as it demonstrates a flatter cost surface than the others. The open source scheme could be the all-round winner when such a choice is available and if the developing team has the proper skills. Model users can narrow down the scope of F and w_2 (or other combinations) in line with their own supply chain environment for fine grain investigations.

5.4 Importance of Early Preparation

According to the results of the previous experiments, it is noticeable that the process share w_1 has a substantial contribution to the final cost. To a SCMS, w_1 is not only meaningful to ready-made software packages (COTS or OSS) but also self-developed systems (Framework or Proprietary). In the latter case, w_1 indicates the share of preliminary development before the supply chain system functioning, if the high frequency processes are already known at the early stage. The significance of early stage preparations (include stages of planning, requirement collection and analysis, system design) has been recognised by most software engineering researches and project management literature (Murch, 2000; Sommerville, 2000; Hedeman et al., 2005). This cost model provides an effective way to assess the weight of w_1 in a quantitative way.

In the following experiment, the influence of w_1 is evaluated in such a design: keep the manual process share constant (i.e. $w_1 + w_2 \equiv 0.9$), increase w_1 from 0 up to 0.9 to inspect the final cost (the bounded w_2 decreases from 0.9 to 0 accordingly). All the other variables remain at average values specified in the value range table (Table 4.1). As the implementation schemes basically implies different set of values of the cost model variables, the conclusion is therefore universally applicable.

The values of model variables for this experiment are combined in Table 5.6:

Table 5.6: Value list for early developing share w_1 OAT analysis

Variable	T_f	F	D	τ	μ_m	μ_p	v
Value	10,000	150	25	100	1.48	0.31	4.5
Variable	x	dc	dc'	C_I	w_1		w_2
Value	0.5	850	2,550	25,000	$[0, 0.1, \cdots, 0.9]$		$[0.9, 0.8, \cdots, 0]$

A matrix of cost results is therefore established by substituting above values into the cost model. The figures are listed in Table 5.7. Figure 5.3 illustrates the result matrix in a stacked bar chart to exhibit the cost transition in an intuitive fashion.

Table 5.7: Early developing share w_1 OAT analysis

w_1	w_2	C_I	C_M	C_S	C_D	C_R	C_X	C (£)
0	0.9	25,000	2,362	77,500	229,500	77,005	91,381	502,748
0.1	0.8	25,000	2,362	49,101	216,750	77,005	55,887	426,105
0.2	0.7	25,000	2,362	30,992	204,000	77,005	33,536	372,895
0.3	0.6	25,000	2,362	19,446	191,250	77,005	19,564	334,627
0.4	0.5	25,000	2,362	12,084	178,500	77,005	10,935	305,886
0.5	0.4	25,000	2,362	7,390	165,750	77,005	5,713	283,220
0.6	0.3	25,000	2,362	4,396	153,000	77,005	2,663	264,427
0.7	0.2	25,000	2,362	2,488	140,250	77,005	999	248,104
0.8	0.1	25,000	2,362	1,271	127,500	77,005	218	233,356
0.9	0	25,000	2,362	495	114,750	77,005	0	219,612

The cost results reveal that the early developing share w_1 has a principal influence on the cost elements of C_X, C_S and C_D. The final cost C, while $w_1 = 0.9$ (namely, process types are fully implemented in the beginning apart from those 10% share of process types left to manual handling), is less than half of the cost compared to the purely upgrade-based development (when $w_1 = 0$). According to the decelerating downward slope of the total cost along the w_1 axis, it is very cost effective to implement those high frequent supply

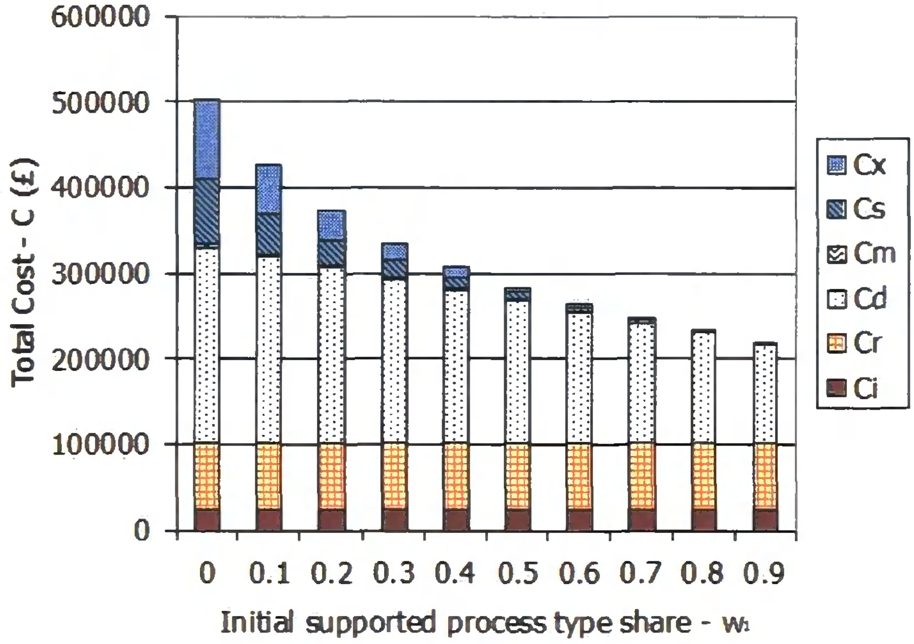


Figure 5.3: Early developing share w_1 OAT analysis bar chart

chain processes at the early stage before SCMS functioning (when $t < 0$). The gradient of cost trend is governed by the process diversity v.

5.5 Influences of Developing Team

The developing team is another important "manageable" aspect to the SCMS planner. Regardless whether the project team comprises of internal, contracted or combined developers, some elements are common in successful SCMS practices. The key points and techniques have been well stated in software engineering literature (Brooks, 1975; Sommerville, 2000). A SCMS, as it still falls in the scope of software products, should be able to inherit these research outcomes directly. Further details in this area are out of this research scope but easily attainable by referring to software engineering research publications, such as the literature mentioned above.

For a complete SCMS implementation guidance, some key points have to be briefly discussed to establish the links between the proposed cost model and traditional software engineering research outcomes.

Experience The experience of a SCMS project team indicates not only technical skills but also knowledge of the particular supply chain environment. Therefore a hired developing team with high technical skills is not necessarily the best choice. The degree of experience can influence model variables of the average process develop time τ and also the unit system operating cost μ_p. A lower μ_p in a well implemented SCMS has remarkable influence to the total cost reduction.

Priority The cost model takes the proper process developing priority by occurrence rate of respective process types as an implementation premise. Failure to follow this condition incurs unnecessary high cost in the end.

Team-size One aspect that deserves careful leverage is the size of the developing team: resource consumption of a developing team (salary, management overheads, computers, workspace, etc.) increases along with its size but not necessary the *productivity*. It is also a waste to maintain a large team on a small scale supply chain project where uncertainty is substantial. Literature research is strongly recommended to achieve a good balance between the team size and its cost.

5.6 Summary

In this chapter the prime SCMS implementation related cost effective analyses, i.e. implementation scheme, developing team size and early stage preparation are performed by exploiting the cost model. In the implementation scheme section, it has been stated that there is no absolute best solution to all SCMS - The total cost is a combined effect of technical, environmental and user-specific conditions.

In order to obtain reasonable outputs from this cost model, SCMS planners need to have a deep understanding of the introduced cost model variables as well as their own supply chain environments. With the provision of this quantitative cost model, planners are encouraged to carry out further experiments in addition to the ones already introduced, to satisfy their own supply chain demands.

Chapter 6

Discussions

This chapter reviews the research work in association with relevant research initiatives discussed in the literature review to show what have been achieved. It also summarises the advantages and limitations that have been discussed throughout this research.

6.1 Summary of the Research

The ultimate intention of this study is to assist in determining the best practice of a supply chain management system from the cost effective point of view. The research results confirmed it was achievable by using appropriate technologies and plans. This study achieved the objective by developing a mathematical temporal cost estimation model which was suitable in the context of supply chain management since commercial software systems were mostly elicited by potential profit returns. This model facilitated the information system cost estimation in the supply chain environment by paying prime attention to its uncertainty and flexibility. This cost model considered the manual work as part of the whole "system" in addition to software applications, therefore a complete projection of the actual benefits against costs over the time was shaped. This study inherited a few ideas and methods from the information system cost model of Gebauer and Schober, but resolved the problem from a different perspective. The primary reasons for designing a new cost model instead of reusing or modifying other cost models in the supply chain context were subjected to the following considerations:

1. Change is the nature and advantage of supply chains. It is not realistic to get the SCMS fully ready in the beginning and keep it invariant afterwards. To obtain a precise estimation, the information system development should be treated as a persistent activity along the system lifetime.

2. This cost model assumes that the system development is progressed in a sequential manner. It is due to the following reasons:

    - Firstly, it is the common developing pattern of bespoke systems since the team size in most cases is limited and fixed. A stable team size is recommended by software engineering researches as change of developing team incurs many management difficulties;

    - Secondly, supply chain related changes occur throughout its whole life time, which turn the system development into a persistent developing loop as the details of changes have to be captured before proceeding software development. This is the prime reason to develop a new cost model instead of inheriting from existing methods.

3. Performance measurement and KPI based approaches are dependent on

personal experience. The complexity increases exponentially in large supply chain environment when cross-referencing with business demands. This leads to the problem of incomparable or meaningless performance indicators. This temporal cost estimation model is a more systematic solution.

This cost estimation model construction was started from a single generic process cost analysis, followed by the multi-process cost composition, where the developing work was arranged for processes one after another. Based on the multi-process cost trend line chart and its mathematical representation, the total cost was split into several cost elements, each of which was reviewed and simplified by means of the Lorenz Curve and mathematical theorems. During the course of model construction, a set of variables were introduced to convert the cost model from discrete process cost composition into statistical abstraction. In this way, the supply chain system planners were relieved from technical prerequisites, or excessive implementation details which were barely known at the early stage of the supply chain projects.

The sole cost estimation model did not provide sufficient guidance on how to achieve cost effectiveness for actual supply chain exercises. To overcome this problem, several analytical experiments were performed to establish hand-on experience of mapping actual supply chain environment onto the model. By identifying the variables from fixed characteristics of a particular supply chain, the cost saving task was turned into a mathematical exercise of finding the minimal value of a multi-variable function (or a single-variable function in simple cases). In addition, to help cost model practice, the cost sensitive variables that are worth planners' extra attention were also assessed in the model analysis sections.

The cost estimation model is useful not only in attaining cost estimation of a determinate SCMS implementation plan, but also in assisting the decision making process by comparing various implementation choices. In the SCMS implementation scheme evaluation chapter, typical schemes were analysed to demonstrate the method in order to find out the suitable one by considering their respective strengths and weaknesses. System planners can follow the analytical method to discover their own appropriate scheme. In general, this study covered the majority of what SCMS planners would need to consider in finding a cost effective SCMS implementation.

For the purpose of early stage SCMS cost estimation, this model was established on several assumptions to remain concise. This inevitably involved several types of deviations which had not been studied thoroughly. The advantages and limitations of this research are outlined in the following sections for SCMS practitioners or researchers to cross-reference, possibly together with other cost estimation techniques, in order to achieve an optimal estimation.

6.2 Advantages

First of all, as a mathematical quantitative model, it is superior in areas of cost estimation compared to empirical performance measurement methods, particularly when there are only few similar research initiatives to cross-reference with (Creswell, 2003). Considering the 13 model variables and their interactions, hundreds or even thousands of guidelines and performance indicators might be needed to achieve the same effect, provided that users are still able to follow. This model is universally applicable since demands of local knowledge or information technology background are minimised.

There is only a small amount of cost model variables for the system planners to balance at the planning stage, in accordance with their own supply chain environment. The model variables (Table 3.1), derived from supply chain process cost composition and statistic abstraction, have pragmatic meanings hence are easy to be assigned. In addition to the cost model, analytical experiments were also performed to address the prime SCMS planning issues before system development.

As a result of the time-dependency of this cost model, the cost trend at each project stage can be investigated for accurate financial arrangement. This cost visibility is a very beneficial feature in reality. After all, the supply chain system host company's major concern is how it achieves savings in the end.

This cost estimation model managed to avoid excessive empirical elements and implementation details as required in typical VBSE approaches (the COCOMO II for instance). This model might not be superior in terms of accuracy but it is more intuitive and practical to apply. In addition, with VBSE approaches, users may get very different results due to the diversity of understanding of empirical factors. This should not occur to the proposed cost estimation model, which mostly encompasses variables with pragmatic meanings.

Compared to the cost model by Gebauer and Schober (2006), this SCMS cost estimation model considers the delay of process development to imitate the real supply chain software implementation schedule instead of plain cost superpositioning. In theory it should result in more accurate results, particularly while differentiation of unit manual process cost and unit machine process cost is remarkable or when the developing work persists for a long period. This temporal characteristic ensures this model to be an appropriate choice in the supply chain context.

6.3 Limitations

The major limitation of this research as a whole, lies in the lack of large amount of industrial case studies to verify its feasibility. Supply chain management system implementation is usually a long term process which requires continuous accounting tracking. It is rather difficult to conduct an effective case study for this type of research. A convincing conclusion from case studies requires not only sufficient amount of studies, but also a wide coverage of various sized SCMS implementations. This, however, requires a long term continuous research.

The case study task becomes more difficult in proving whether the cost model suggests the best choice on implementation scheme, as a result of the HERACLITUS'S LAW[1]:

> *"You cannot step into the same river twice, for fresh waters are ever flowing in upon you."*

In the supply chain context, this law means that a SCMS cannot be practiced more than once. This is due to the fact that the supply chain environment evolves all the time which prevents different implementations being tested under the same conditions, provided the commercial feasibility is positive. The actual cost is also influenced by people's knowledge and experience learned from previous exercises. Therefore, the method of comparing different implementations in the same supply chain environment is not feasible.

Another possible way to tackle the problem is by conducting different SCMS implementations on several *similar* supply chain practices for comparison. Unfortunately, there is still no clear definition on similarity of supply chains so far.

[1] An ancient Greek philosopher who came after Pythagoras, before Socrates.

The comparison based research among different supply chain organisations is therefore also pointless at the current stage. This problem may be avoidable in the future when more research initiatives and SCMS case studies emerge in this research area. Likewise, this temporal cost model can be refined with findings from similar researches on account of the reciprocal influences.

Back to the cost estimation model itself, it has been explained during the model reasoning that there are several cost deviations sourced from statistical simplifications and assumptions at various stages. They are summarised here for future reference.

This cost model is constructed on the premise that the system development is ordered by process occurrence frequency. This order achieves the best cost-effectiveness in theory but is hard to follow completely in real practices as the process frequencies may not always be known to the software developers at that time. Also the frequencies of a process may not stay invariant. The actual cost will increase due to the wrong developing order. It might be useful to introduce an empirical coefficient in the equation to rectify the mis-ordering deviation, but substantial case studies are essential in establishing the coefficients matrix for various SCMSs.

Subject to the limitation that this cost model cannot apply respective developing duration τ on FTC and FTU processes, the only place in the cost model to specify the difference is merely the developing cost per process. In fact, FTC processes should take longer to develop than FTU since the influence of FTC is module or infrastructure wide by definition. During the course of FTC development, the longer developing time is, the more process tasks have to be handled manually. This delays the schedule of all the following process implementations as well. All these consequences result in extra costs eventually. The influence of consequences is still to be quantified. A better statistical simplification that bypasses this problem could be one of the future research directions.

Chapter 7

Conclusions

In this study, it has been proved that the supply chain management system implementation cost can be estimated, despite the high uncertainty and flexibility of supply chains. The prime challenges at the SCMS planning stage have been resolved by a mathematical cost model, with associated system implementation analysis and discussion. The SCMS planners should be able to cross-reference the benefits against costs to make appropriate decisions by means of this model. In addition, SCMS planners can also assess the feasibility, scale and implementation scheme of their own SCMS without much difficulty.

As already stated in the literature review, the majority of information system cost estimation methods either require excessive details and blurred adjustment factors, or lack solid evidences of the correlation between individual practices and the empirical guidelines. The steep demand of experience and technical skills also obstructs the successful application of those methods. This study has constructed a new temporal cost estimation model that can satisfy the objectives without the shortfalls of both empirical guideline-based and software engineering-centric approaches. This cost model adheres to the practicality by focusing on the coarse characteristics of a supply chain. The weak relation with information technologies also benefits the feasibility and durability of this cost model.

During the model analysis several important findings have been addressed for cost-effective practices. These findings should match existing software engineering research outcomes, in areas like system architecture or plenary development. Nevertheless, the intrinsic advantage of this cost model is that it provides a quantitative method to measure the importance of these rules or guidelines from the costing aspect, which facilitates the project planning and management to a substantial extent.

This study is a step forward towards sophisticated supply chain management system construction. With no doubt there are still areas to be refined. This, however, depends on availability of resources and in-depth surveys.

References

Adamic, L. A. (2000), Zipf, Power-Laws and Pareto - a ranking tutorial, Tutorial, Information Dynamics Lab, HP Labs.

Applegate, L. M., McFarlan, F. W. and McKenney, J. L. (1999), *Corporate Information System Management: The Challenges of Managing in an Information Age*, 5 edn, Boston, MA:Irwin/McGraw-Hill.

Barrentine, L. B. (1999), *An introduction to design of experiments: a simplified approach*, Milwaukee : ASQ Quality Press.

Basmann, R. L., Hayes, K. J., Slottje, D. J. and Johnson, J. D. (1990), 'A general functional form for approximating the Lorenz Curve', *Journal of Econometrics* **43**, 77–90.

Batchelor, R. (1994), *Henry Ford, Mass Production, Modernism and Design*, Manchester University Press ND, chapter 5, pp. 124–143.

Benediktsson, O., Dalcher, D., Reed, K. and Woodman, M. (Henry Ford, Mass Production, Modernism and Design), 'COCOMO-based effort estimation for iterative and incremental software development', *Software Quality Journal* **11**, 265–281.

Boehm, B. (1981), *Software Engineering Economics*, Prentice-Hall.

Boehm, B. (2003), 'Value-based software engineering', *Software Engineering Notes* **28**(2), 3–15.

Boehm, B., Abts, C. and Chulani, S. (2000), 'Software development cost estimation approaches - a survey', *Annals of Software Engineering* **10**(1-4), 177–205.

Boehm, B. and Royce, W. (1989), Ada COCOMO and the Ada process model, *in* 'Fifth COCOMO Users' Group Meeting', Storming Media LLC.

Boehm, B. W., Steece, B. and Madachy, R. (2000), *Software Cost Estimation With COCOMO II*, Prentice-Hall.

Brooks, F. P. (1975), 'The mythical man-month', *SIGPLAN Notices* **10**(6), 193.

Byrd, T. A. and Turner, D. E. (2000), 'Journal of management information systems', *Measuring the Flexibility of Information Technology Infrastructure: Exploratory Analysis of a Construct* **17**(1), 167–208.

Cacuci, D. G. (2003), *Sensitivity and Uncertainty Analysis: Theory*, Vol. 1, Chapman & Hall.

Chen, I. J. and Paulraj, A. (2004), 'Understanding supply chain management: critical research and a theoretical framework', *International Journal of Production Research* **42**(1), 131–194.

Chopra, S. and Meindl, P. (2004), *Supply Chain Management: Strategy, Planning and Operation*, 2 edn, Pearson Prentice Hall, chapter 1, pp. 1–43.

Chotikapanich, D. (1993), 'A comparison of alternative functional forms for the Lorenz Curve', *Economic Letters* **41**(2), 129–138.

Cosier, J. P. (1987), *Variation Research (Statistical Engineering) Workshop Manual*, revised edn, Philips N.O.R.G Blackburn.

Creswell, J. W. (2003), *Research Design: Qualitative, Quantitative, and Mixed Method Approaches*, 2 edn, SAGE Publications Inc., chapter 8, pp. 142–152.

Czitrom, V. (1999), 'One-factor-at-a-time versus designed experiments', *The American Statistician* **53**(2), 126–131.

Daniel, C. (1973), 'One-at-a-time plans', *Journal of the American Statistical Association* **68**(342), 353–360.

Disney, S. M., Naim, M. M. and Potter, A. (2004), 'Assessing the impact of e-business on supply chain dynamics', *International Journal of Production Economics* **89**(2), 109–118.

Fisher, R. A. (1935), *The Design of Experiments*, Oliver & Boyd Edinburgh, Scotland.

Ford, H. (1926), *Today and Tomorrow*, Portland, OR: Productivity Press, Inc.

Ford, H. and Crowther, S. (2005), *My Life and Work*, Gutenberg eBook, chapter 4, pp. 64–77.

Gastwirth, J. L. (1971), 'A general definition of the Lorenz Curve', *Econometrica: Journal of the Econometric Society* **39**(6), 1037–1076.

Gebauer, J. and Schober, F. (2006), 'Information system flexibility and the cost efficiency of business processes', *Journal Of The Associaton for Information Systems* **74**(3), 122–147.

Gehani, N. (1983), *Ada: An Advanced Introduction*, Prentice-Hall.

Graham, D. R. (1989), 'Incremental development: Review of nonmonolithic life-cycle development methods', *Information and Software technology* **31**(1), 7–20.

Gunasekaran, A. and Ngai, E. W. T. (2004), 'Information systems in supply chain integration and management', *European Journal of Operational Research* **159**, 269–295.

Gunasekaran, A., Patel, C. and Tirtiroglu, E. (2001), 'Performance measures and metrics in a supply chain environment', *International Journal of Operations & production Management* **21**, 71–87.

Gunther, R., Levitin, L., Shapiro, B. and Wagner, P. (1996), 'Zipf's law and the effect of ranking on probablity distributions', *International Journal of Theoretical Physics* **35**(2), 395–417.

Hammer, M. and Champy, J. (1993), *Reengineer the Corporation*, New York, NY:Harper Collins.

Hannus, T., Poignee, O. and Schiefer, G., eds (2003), *The implementation of a web based supply chain information system - experiences with a reginal quality grain program*, University of Bonn, EFITA 2003 Conference.

Hanseth, O., Monteiro, E. and Hatling, M. (1996), 'Developing information infrstructure: the tension between standardisation and flexibility', *Science, Technology and Human Values* **11**(4), 407–426.

Hedeman, B., van Heemst, G. V. and Fredriksz, H. (2005), *Project Management Based on PRINCE2*, Van Haren Publishing.

Helton, J. C., Johnson, J. D. and Salaberry, C. J. (2006), 'Survey of sampling based methods for uncertainty and sensitivity analysis', *Reliability Engineering and System Safety* pp. 1175–1209.

Hugos, M. (2003), *ESSENTIALS of Supply Chain Management*, John Wiley & Sons, Inc., chapter 1, pp. 1–42.

Johnson, M. E. and Whang, S. (2002), 'e-business and supply chain management: An overview and framework', *Production and Operations Management* **11**(4), 413–423.

Kakwani, N. C. and Podder, N. (1976), 'Efficient estimation of the Lorenz Curve and associated inequality measures from grouped data', *Econometrica* **44**, 137–148.

Kauffman, R. J. and Walden, E. A. (2001), 'Economics and electronic commerce: Survey and research directions', *International Journal of Electronic Commerce* pp. 5–117.

Lambert, D. M., James, R. and Ellram, L. M. (1998), *Fundamentals of Logistics Management*, Boston, MA:Irwin/McGraw-Hill, chapter 14, pp. 504–546.

Lee, H. L. (2004), 'The triple - a supply chain', *Harvard Business Review* **82**(10), 102–115.

Lee, H. L., Padmanabhan, V. and Whang, S. (1997), 'Information distortion in a supply chain: The bullwhip effect', *Management Science* **43**(4), 546–558.

Lehman, M. M. (2000), 'Rules and tools for software evolution planning and management', *FEAST2000* pp. 53–68.

Lorenz, M. O. (1905), 'Methods of measuring the concentration of wealth', *American Statistical Association* **9**, 209–219.

Milgate, M. (2001), 'Supply chain complexity and delivery performance: an international exploratory study', *Supply Chain Management: An International Journal* **6**(3), 106–118.

Morgan, C. (2004), 'Structure, speed and salience : performance measurement systems design : a literture review and research agenda', *Business Process Management Journal* **10**(5), 522–558.

Muckstadt, J. A. (1997), A paradigm lost, Technical Report 1180, School of Operations Research and Industrial Engineering, Cornell University, Ithaca, NY.

Murch, R. (2000), *Project Management: Best Practices for IT Professionals*, Prentice Hall PTR.

Murillo, L. (2001), 'Supply chain management and the international dissemination of e-commerce', *Industrial Mangement & Data Systems* **101**(7), 370–377.

Neely, A., Gregory, M. and Platts, K. (1995), 'Performance measurement systems design: a literature review and research agenda', *International Journal of Operations & production Management* **15**(4), 80–116.

Oakley, J. E. and O'Hagan, A. (2004), 'Probabilistic sensitivity analysis of complex models: a Bayesian approach', *Journal of the Royal Statistical Society. Series B* pp. 751–769.

Oliver, R. K. and Webber, M. D. (1992), 'Supply-chain management : logistics catches up with strategy', *Logistics: The Strategic Issues* pp. 63–75.

Ortega, P., Martin, G., Fernandez, A., Ladoux, M. and Garcia, A. (1991), 'A new functional form for estimating Lorenz Curves', *Review of Income and Wealth* **37**, 447–452.

Parmenter, D. (2007), *Key Performance Indicators*, John Wiley & Sons, Inc., chapter 1, pp. 3–10.

Pigou, A. C. (1912), *Wealth and Welfare*, Mcmillian and Co., Limited.

Plackett, R. L. and Burman, J. P. (1946), 'The design of optimum multifactorial experiments', *Biometrika* **33**(4), 305–330.

Premkumar, G. P. (2000), 'Interorganization systems and supply chain management: an information processing perspective', *Information Systems Management* **17**(3), 56–69.

Quinn, F. J. (1997), 'What's the buzz?', *Logistics Management* **36**(2), 43–50.

Reed, W. J. (2001), 'The Pareto, Zipf and other power laws', *Ecomonic Letters* **74**(1), 15–19.

Saltelli, A., Chan, K. and Scott, E. M. (2004), *Sensitivity analysis in practice: a guide to assessing scientific models*, John Wiley & Sons, Inc., chapter 2, pp. 47–87.

Shepherd, C. and Gunter, H. (2005), 'Measuring supply chain performance: current research and future directions', *International Journal of Productivity and Performance Management* **55**(3/4), 242–258.

Sommerville, I. (2000), *Software Engineering*, 6 edn, Addison Wesley.

Talluri, S. (2000), 'An IT/IS acquisition and justification model for supply-chain management', *International Journal of Physical Distribution and Logistics Management* **30**, 221–237.

Trigeorgis, L. (1993), 'Real options and interactions with financial flexibility', *Financial Management* **22**(3), 202–225.

Upton, D. M. (1994), 'The management of manufacturing flexibility', *California Management Review* **36**(2), 72–90.

Williams, T. and Kelley, C. (1993), *GNUPLOT: An Interactive Plotting Program*.

Appendix A

GNUPLOT Source Code For Figure 5.2

Gnuplot (Williams and Kelley, 1993) is a program with a long history started from 1986. It can generate two- and three-dimensional plots of functions and data fits. The source code to generate the 3D chart of Figure 5.2 is as follow (omit the line numbers if run in gnuplot):

```
##################################################
# 3dplot.plt                                     #
# code to draw the 3D diagram of the cost model #
##################################################
set samples 30
set isosamples 30
set hidden3d
#set title "3D cost graph of w2 and F"
set view 60,30
set xlabel "w2"
```

```
set ylabel "F"
set xrange [0.1 : 0.9]
set yrange [10 : 500]
#set autoscale z
Um = 1.48
D = 25
v = 4.5
Tf = 10000
L(w) = (exp(w * v) - 1)/(exp(v) - 1)
Cm(w1, w2) = L(1 - w1 - w2) * D * Tf * Um
Cs(w1, Up) = L(1 - w1) * D * Tf * Up
Cd(F, w1, w2, xx, dc, dc1) = F * w2 * xx * dc1 + F * (w1 + w2 * (1 - xx)) * dc
Cr(w1, w2, Up) = (1 - L(1 - w1 - w2)) * D * Up * Tf
p1(w1) = exp(v * (1 - w1)) / (exp(v) - 1)
p2(w2, F) = (1 - exp(-v * w2)) / (1 - exp(-v / F))
p3(w2, F) = F * w2 * exp(-v * w2)
Cx(w1, w2, F, Up, t) = D * (Um - Up) * t * p1(w1) * (p2(w2, F) - p3(w2, F))
C(Ci, w1, w2, Up, F, xx, dc, dc1, t) = Ci + Cm(w1, w2) + Cs(w1, Up) + Cd(F, w1, w2, xx, dc, dc1) + Cr(w1, w2, Up) + Cx(w1, w2, F, Up, t)

#COTS
C1(x,y)=C(26000, 0.4, x, 0.28, y, 0.8, 1000, 3000, 120)
#Framework
C2(x,y)=C(12500,   0, x, 0.31, y, 0.3,  700, 2100, 80)
```

```
#Proprietary
C3(x,y)=C( 6000,    0, x, 0.31, y, 1.0,  0.1, 1500, 100)
#Open-Source
C4(x,y)=C( 6000, 0.2, x, 0.34, y, 0.8,  850, 2550, 100)
set terminal png
set output "3dplot.png"
splot C1(x, y),C2(x, y),C3(x, y),C4(x, y)
```

Appendix B

Paper Published in HAAMAHA 2007 Conference Proceedings

A cost model for early design of supply chain information systems

ABSTRACT

In supply chain (SC) practices, Information System (IS) plays a crucial role in characterising its capability, efficiency as well as economy. However, as a result of the high uncertainty of supply chain, it is a challenge itself to plan the information system properly with regard to infrastructure, resources and in particular - cost estimation.

The majority of IS cost models introduced in software engineering research relies on determinate use cases which SC can not satisfy. This paper provides a theoretical model that can help to estimate the overall IS cost for decision making purpose.

Keywords

Supply Chain, Information System, Cost Model, Early estimation, Lorenz Curve

INTRODUCTION

Background

Supply Chain, as a successor of logistic network towards the global trade popularization, attracted public and research attentions and came into pervasive practice in the late 1990s [8,13]. Supply chain enables its participators to coordinate closely through in-depth information sharing to facilitate prompt interactions and minimise transaction overheads [12]. Information system is addressed as the key driver that connects various fields of supply chain [5]. Therefore a suitable information system is essential towards a successful SC practice.

Different from general purpose software packages, Supply Chain Information Systems (SCIS) have to be designed individually to satisfy "unique" and "evolutional" requirements sourced from combination of various supply chain partners, policies, procedures and changes. Commercial-Off-The-Shelf (COTS) software products and turnkey systems can be considered as entry points if their flexibility and extensibility are sufficient for future extension.

Treating these entry points as infrastructure, SCIS planner has to evaluate to what extent it can be upgraded to cope with supply chain complexity, uncertainty and volatile collaboration [11] with low cost. This is regarded as key characteristics of system flexibility [1]. According to definitions given in [7], the term flexibility denotes either: (A) flexibility to changes (FTC) or (B) flexibility to use (FTU). In this research, the system upgrade is defined as creating new component from scratch or replacing existing component that is already in use by other processes as the boundary to distinguish these two flexibility categories. Accordingly, flexibility to change and use are relative to the SCIS infrastructure capability. A good system infrastructure can maximise FTU and keep "unsupported processes" in low scale.

Despite the significance of flexibility, it is merely a resort towards cost effectiveness which is really this paper concerns. So far, SCIS cost estimation hasn't drawn many research attentions. The SCOR[1] benchmark, derived from its supply chain tiered model,

[1] Supply Chain Council, a commercial supply chain research group founded from industrial members. Url: http://www.supply-chain.org

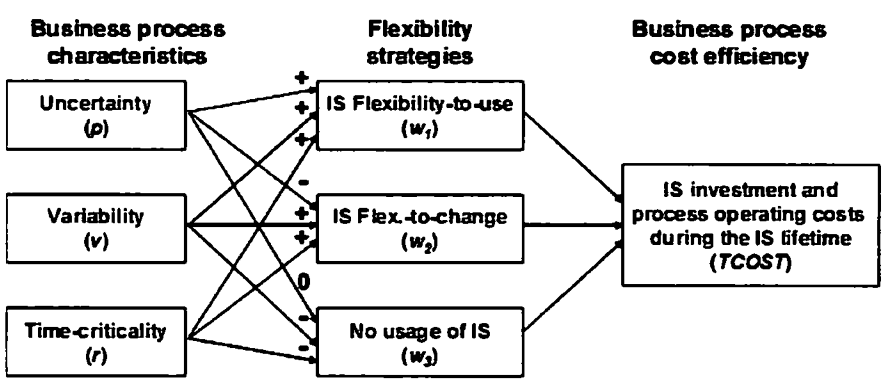
pays full attention to strategic supply chain performance review and is hardly useful in SCIS cost estimation. Traditional software engineering cost estimation initiatives such as COCOMO 2.0 and its predecessor [2, 3], employ too many empirical adjustments that are very difficult to use and assess the result without large scale case studies.

Gebauer's Research

Gebauer and Schober [6] proposed a theoretical information system cost model that emphasises on flexibility and variability. Although this model doesn't specifically target to supply chain environment, its process oriented view presents a suitable ground work for this research.

Figure 1 illustrates the decision routes that are used to determine the cost efficient mix of flexibility strategies in support of an arbitrary business process. Briefly, a process request is handled either by computer or manually, when the process type isn't supported by SCIS. Next, if it is possible and necessary to incorporate this process into SCIS, the software upgrade will happen either by extending the existing component (FTU) or by creating or replacing the whole function module (FTC), depends on actual SCIS capability. It is likely that a certain amount of process might be left outside the information system by manual handling for various reasons. The total cost is summarized depends on proportion of occurrence to each process route.

Figure 1. Gebauer's process model [6]



The major contribution of Gebauer's model [6] is that it managed to present a practical cost model from its unique process view with concentration on flexibility and variability. Furthermore, another key successfulness originated in [6] is the usage of cumulative distribution statistical function - Lorenz curve [10] to synthesize and measure the process variability in a quantitative way. With this technique, process variability is described by a single variable – the curvature. Its utilisation greatly benefits the model simplicity, and hereby will be inherited in this research.

At the mean time, a few shortcomings, as by-products of its simplicity, are also revealed. Firstly, from reality, it is clear that information assets (both hardware and software) always have limited lifetime, this lifetime should be taken into account at the planning stage to draw the boundary of flexibility and ultimately to achieve cost effectiveness. Gebauer's model neither defines the system lifetime explicitly nor allows users to specify it as a variable. Secondly and more important, this model is established on a relative static system configuration, the deviation could be tremendous when applying it on dynamic supply chains where system development activities and requests for change happen frequently. The time dimension has to be considered in supply chain cost estimation. It is essential to recognise that the point of time each process getting developed has considerable impacts on the final cost. These limitations are what this research tries to overcome.

In the following section, the SCIS cost model is derived by stepwise equation reasoning with detailed explanations. Following the cost model, the model analysis section investigates the results from various dimensions to summarise several findings and propositions that might be relevant to supply chain practices. Finally, a brief conclusion is given with suggestions for future research.

COST MODEL

Model concept

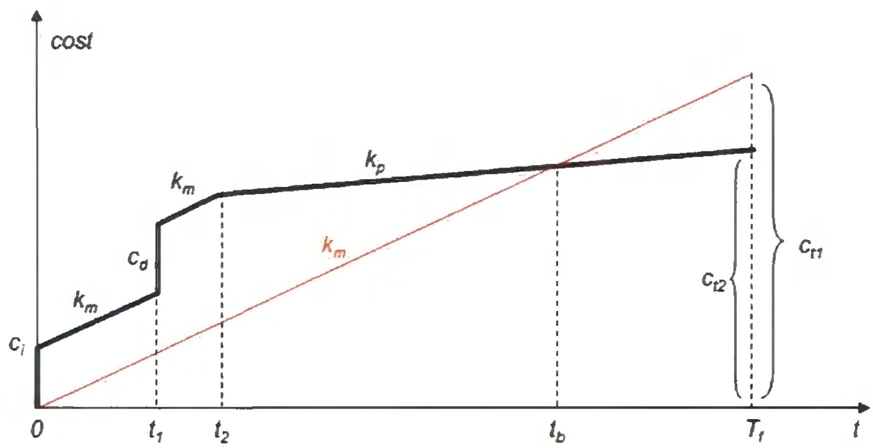
Gebauer's abstraction of process handling workflow (Figure 1) founds the entry point of this research. Regardless of whether the supply chain information system is made up of commercial off-the-shelf products or in-house developed kits, it is safe to claim that more or less of the system will go through a reconfiguration and developing circle during its life time to adapt to changes of supply chain. Therefore, this model asserts the unit total cost of any supply chain process comprises of initial cost, manual handling cost, software development cost and system process cost.

The initial cost could come from computer hardware, software purchasing as well as extra labour charge. In real case, it is up to the decision maker to balance the share of investment that could also be used for other purposes. For instance, computer server hardware is purchased to host the SCIS but other systems can also be installed into it to exploit the full capacity, in this case the share of SCIS hardware cost is basically a financial trade-off. As this kind of cost is usually countable at early stage, it is simply denoted as initial cost in this cost model which concerns more on the cost growth. Software development cost is treated as a lump sum for simplicity because no matter how the money is spent in reality, the one-off sub total remains. Manual handling cost is simply regarded as a linear function to transaction count by employing a constant of unit manual handling cost per transaction. Similar assumption may be applied to unit system process cost.

Unit process cost analysis

When a supply chain process request arrives, if it is yet not in the information system coverage, manual workflow takes over for the time being. Software designers evaluate the necessity and possibility of automating this process type thereafter. Once decision is made to proceed the development, budget will be spent to drive the development. At the meantime this kind of process is still tackled manually until the upgrade is in place. Attributes to the replacement of manual handling with system automation, the cost growth slows down towards the end of system life time. The cost line chart can be plotted as Figure 2.

Figure 2. Unit process cost line chart



In this chart, the bold black line outlines the estimated cost to the end of system life time T_f. c_i stands for initial cost and c_d for development cost respectively. k_m with its counterpart k_p, are defined as slopes that denote cost spent per unit time via manual handling and system automation, therefore $k_p < k_m$ (otherwise leaving the process type to manual work is more sensible).

The red diagonal drawn in Figure 2 is the manual cost line for reference purpose. After breakeven point t_b, overall cost c_{t2} goes lower than manual cost c_{t1}. In addition, this chart also indicates manual cost is more cost effective when $T_f \le t_b$. This could be one of the reference points to decide whether this process type should be implemented in SCIS. Equivalent mathematical presentations to Figure 1 are as follows:

$$c_{t1} = k_m \cdot T_f \qquad \textbf{(E1)}$$

$$c_{t2} = c_i + c_d + k_m \cdot t_2 + k_p \cdot (T_f - t_2), \text{ where } k_p < k_m \qquad \textbf{(E2)}$$

Compound process cost

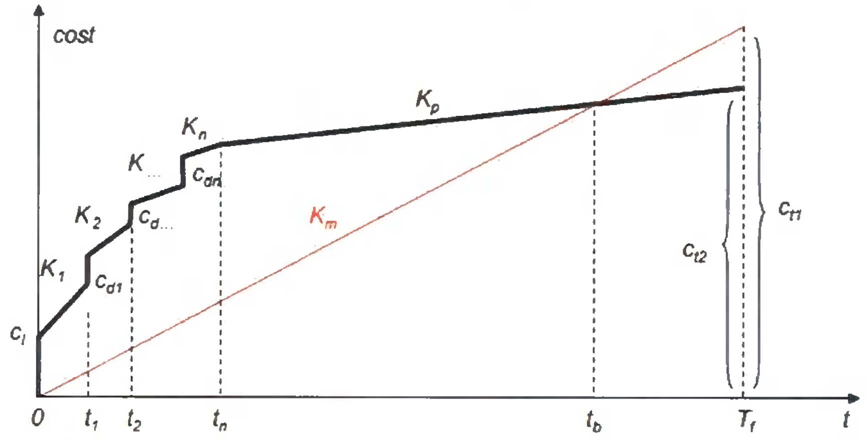
A supply chain information system comprises of multiple processes, some of which are already implemented in the beginning as build-in functions of COTS, or by early stage development. On account of high uncertainty of supply chain, a large proportion of processes will be developed as upgrades on existing SCIS on a sequential basis in terms of business strategy and partner environment transition.

As a fundamental difference to Gebauer's model [6], this cost model treats the development procedure in a sequential manner with respects to system life time rather than a plain summation to all process types (i.e. parallel development mode). This is due to the fact that in reality, no matter whether the supply chain system is implemented in-house or by contracting to software vendors, the project working group maintains steady productivity at the most of the time. With these external constrains and possible process inter-dependencies, this cost model opts the sequential development as premise to achieve more reasonable cost estimation.

It is also essential to realise that each process type has its unique process transaction rate. Take order process as an example, suppose a company receives orders from partner A for 100 a day but only gets 10 from partner B in different format, then it is obvious that automating process-A prior to process-B is much more cost effective.

Process-B could be left to manual handling at all if it turns out $c_{t1} \le c_{t2}$. This cost model assumes that development activities are organized in optimized fashion by prioritising developing order based on process transaction rate. Political choices are not considered, neither wrong developing priority arrangements subject to lack of knowledge. The conceptual compound cost chart is illustrated in Figure 3.

Figure 3. Compound process cost line chart



Suppose process transactions happen in random order: if the corresponding process type is already supported, then software system takes it over, otherwise, this transaction is handled manually. Therefore, following equations can be established:

$$C_{t1} = K_m \cdot T_f \text{, where } K_m = \sum_{i=1}^{n} k_{mi} \qquad \textbf{(E3)}$$

$$C_{t2} = C_I + (c_{d1} + c_{d2} + \cdots + c_{dn}) + K_1 \cdot t_1 + K_2 \cdot (t_2 - t_1) + K_3 \cdot (t_3 - t_2) + \cdots + K_p \cdot (T_f - t_n)$$

Where $K_x = \sum_{i=1}^{x-1} k_{pi} + \sum_{i=x}^{n} k_{mi}$.

k_{pi}, k_{mi} denote process cost per unit time for process type i handled by information system and manually respectively. c_{di} stands for system development cost for process type i. Substituting K_x into the expression of C_{t2}, the following equation can be derived:

$$C_{t2} = C_I + (c_{d1} + c_{d2} + \cdots + c_{dn}) + (k_{p1} + k_{p2} + \cdots + k_{pn}) \cdot T_f + \sum_{i=1}^{n} (k_{mi} - k_{pi}) \cdot t_i \qquad \textbf{(E4)}$$

In addition to cost ingredients in equation E4, there would be some processes get implemented in the beginning ($t = 0$). We detach this operational cost from C_I and denoted it as

$$C_S = K_S \cdot T_f \text{, where } K_S = k_{sa} + k_{sb} + k_{sc} + \cdots$$

On the other hand, some processes never get developed in system within the whole system life time T_f as a result of their low transaction share. These manual transactions introduce the operation cost of

$$C_M = K_M \cdot T_f \text{, where } K_M = k_{ma} + k_{mb} + k_{mc} + \cdots$$

Finally, the total cost can be expressed as

$$C = C_M + C_S + C_{t2} \qquad \textbf{(E5)}$$

Model refinement

Process transaction shares and Lorenz curve representation

Equation E5 does give the actual total cost but is hardly useful to early stage cost estimation when most of processes are unknown. To refine this model, it is necessary to notice that process types and transaction share for each type are two different concepts: Process type introduces one-off development cost c_{di}, while transaction share contributes to the day-to-day operation cost. This cost model applies Lorenz curve to map the relation of these two aspects due to its similarity to the classical population - income share proposition [10] that Lorenz curve attempts to characterise. Apply on to this model, it can be denoted the relation as

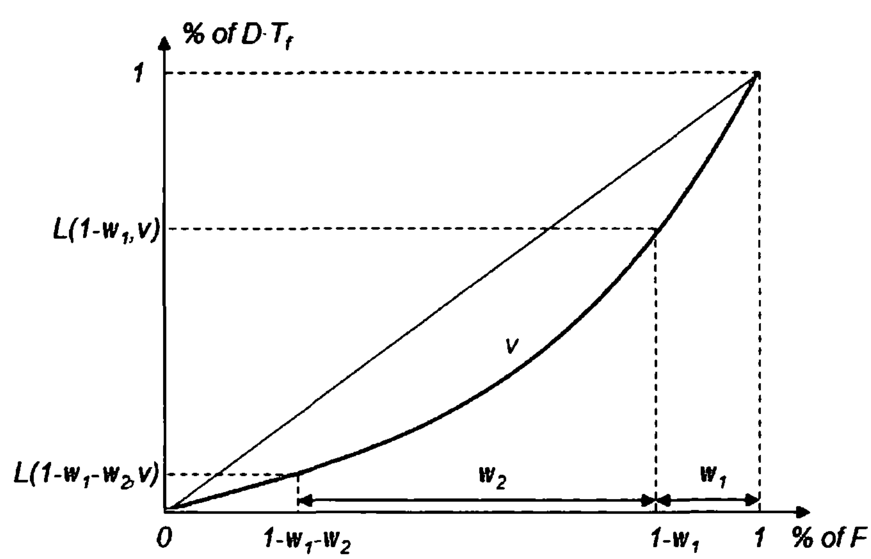
$$y = L(w, v)$$

This function indicates w proportion of process types share y percentage of total transactions that will happen in the proposed SCIS life time, while process types are arranged in incremental frequency order. v is the parameter that indicates the Lorenz curve curvature.

To achieve closest result from application of Lorenz curve against actual cost, we assume the actual system development is optimized in the underlining conditions (visualized in Figure 4):

1. Process types with highest frequency (share of w_1) are implemented at system initial stage.
2. Medium frequency process types are developed after supply chain system goes online in the order of their transaction frequency. Higher frequent process type receives higher development priority. This process proportion is denoted as w_2
3. Only process types with lowest frequency are left to manual handling. $(1 - w_1 - w_2)$ represents this share.

Figure 4. Process type-transaction Lorenz curve



The Lorenz curve mathematical representation introduced in [15] is employed to facilitate future analysis. That is to say,

$$L(w,v)=\frac{e^{v\cdot w}-1}{e^{v}-1}$$, where curvature range is limited to $v\in[0.01,20]$

Cost model simplification

To provide a practical cost estimation model by means of Lorenz curve, each cost ingredient will be revisited to compose the simplified formula. A few extra parameters are presented prior to further analysis.

1. In equation E4, as process transactions are discrete, arbitrary k_{pi} and k_{mi} comply to forms of

$$\begin{cases} k_{pi}=\mu_{pi}\cdot f_i \\ k_{mi}=\mu_{mi}\cdot f_i \end{cases}$$

   Where μ represents unit cost to single transaction; f_i denotes transaction frequency of process type i.
2. F is defined as count of total process types
3. This model assumes process transaction density maintains constant in the main, denoted as D for process transaction count per unit time. Consequently, total transaction count in system life time can be marked as $D\cdot T_f$.
4. Assume share x of process types fit to flexible-to-change category in the total processes that implemented as upgrade (namely within $F\cdot w_2$).
5. Software development introduces lump-sum average unit cost dc to FTU processes, and a separate dc' to FTC processes.

Ground on above definitions and Figure 4, cost elements C_M and C_S can be expressed as:

$$C_M=K_M\cdot T_f=L(1-w_1-w_2,v)\cdot D\cdot T_f\cdot\mu_m \quad \textbf{(E6)}$$

$$C_S=K_S\cdot T_f=\left(1-L(1-w_1,v)\right)\cdot D\cdot T_f\cdot\mu_p \quad \textbf{(E7)}$$

The system development cost $(c_{d1}+c_{d2}+\cdots+c_{dn})$ in Equation E4 is defined separately as C_D. If the initial process development is considered as part of this cost ingredient (unit process type development cost would be dc), then

$$C_D=\sum_{i=1}^{F\cdot(w_1+w_2)}c_{di}=F\cdot w_2\cdot x\cdot dc'+F\cdot\left(w_1+w_2\cdot(1-x)\right)\cdot dc \quad \textbf{(E8)}$$

System operational cost

$$C_R=\sum_{i=1}^{F\cdot(w_1+w_2)}k_{pi}\cdot T_f=\left(1-L(1-w_1-w_2,v)\right)\cdot D\cdot\mu_p\cdot T_f \quad \textbf{(E9)}$$

The last contribution of cost is from the developing stage where manual handling is gradually replaced by system upgrade. This cost is denoted as

$$C_X=\sum_{i=1}^{F\cdot w_2}\left(k_{mi}-k_{pi}\right)\cdot t_i$$

In most cases, as mentioned earlier, the development team maintains a steady productivity. This fact results in a proportional process development time $(t_i - t_{i-1})$ to developing difficulty of process type i. Since developing cost dc and dc' has been averaged to arbitrary process type in addition with the fact that the cost generally reflects difficulty, $(t_i - t_{i-1})$ is consequently treated as a constant. The average unit develop time is therefore denoted as $\tau = (t_i - t_{i-1})$.

The parameter t_i represents the time gap from system start time $t = 0$ to the point that process type i has its turn to be developed. The precise t_i can only be ascertained when t_1, t_2, ... , t_{i-1} are known. These details are obviously not available at early estimation stage. However, as average developing cost dc and dc' has already been utilised, it is viable to define average developing time τ and τ' under the premise that developing team maintains a steady productivity - the same cost indicates the same code size, which again leads to same developing time if productivity remains constant [3,15].

This model prioritises the developing order by process transaction ratio to comply with Lorenz curve prerequisite. Therefore whether the process being developed is in FTU or FTC category is unknown to the cost model. This problem prevents applying appropriate τ on process types so that τ and τ' have to be treated as the same value, namely $\tau = \tau' = (t_i - t_{i-1})$. This hypothesis undoubtedly introduces deviation to the final cost and might be improved by future study. According to above analysis, C_X could be reformed as:

$$C_X = \sum_{i=1}^{F \cdot w_2} (\mu_m - \mu_p) \cdot f_i \cdot \tau \cdot i$$

After applying Lorenz curve into above equation, the minimal C_X can be obtained as,

$$C_X = \sum_{i=1}^{F \cdot w_2} \left(L(1 - w_1 - \frac{i-1}{F}, v) - L(1 - w_1 - \frac{i}{F}, v) \right) \cdot D \cdot (\mu_m - \mu_p) \cdot \tau \cdot i$$

$$C_X = D \cdot (\mu_m - \mu_p) \cdot \tau \cdot \frac{e^{v \cdot (1 - w_1)}}{e^v - 1} \cdot \left(1 + e^{-\frac{v}{F}} + e^{-\frac{2 \cdot v}{F}} + \cdots + e^{-\frac{(F \cdot w_2 - 1) \cdot v}{F}} - F \cdot w_2 \cdot e^{-v \cdot w_2} \right)$$

From finite geometric progression law, it is known that

$$S_n = 1 + q + q^2 + q^3 + \cdots + q^n = \frac{1 - q^{n+1}}{1 - q}$$

Let $q = e^{-\frac{v}{F}}$, $n = F \cdot w_2 - 1$, the following expression can be derived:

$$1 + e^{-\frac{v}{F}} + e^{-\frac{2 \cdot v}{F}} + \cdots + e^{-\frac{(F \cdot w_2 - 1) \cdot v}{F}} = \frac{1 - \left(e^{-\frac{v}{F}} \right)^{F \cdot w_2}}{1 - e^{-\frac{v}{F}}} = \frac{1 - e^{-v \cdot w_2}}{1 - e^{-\frac{v}{F}}}$$

$$C_X = D \cdot (\mu_m - \mu_p) \cdot \tau \cdot \frac{e^{v \cdot (1 - w_1)}}{e^v - 1} \cdot \left(\frac{1 - e^{-v \cdot w_2}}{1 - e^{-\frac{v}{F}}} - F \cdot w_2 \cdot e^{-v \cdot w_2} \right) \qquad \textbf{E10)}$$

Finally, all these cost ingredients can be summarized to define the total cost

$$C = C_M + C_S + C_D + C_R + C_X \qquad \textbf{(E11)}$$

Parameters required are summarised in Table 1.

Table 1. Cost model parameter list

Parameter	Explanation
C_I	Initial cost to establish infrastructure, developing team, preparation training, etc.
μ_m	Average unit transaction cost for manual process
μ_p	Average unit transaction cost for system supported process
w_1	Process type share that implemented prior to system go-live stage
w_2	Process type share that implemented at system functioning stage
x	Share of uncertain processes that developed as upgrade (within $F \cdot w_2$)
D	Total process transaction frequency (total transactions per unit time)
v	Curvature of process type-transaction share Lorenz curve
T_f	Planned supply chain information system life time
F	Total process types
dc	Average flexible-to-use process development unit cost
dc'	Average flexible-to-change process development unit cost
τ	Average unit process development time period

MODEL ANALYSIS

Given the cost model, it is now possible to evaluate supply chain system build criteria regarding flexibility in terms of supply chain circumstances. In this section, parameters related to flexibility, process distribution and difficulty will be progressively changed to observe their independent impact to final cost. Several propositions will be brought forward to constitute a guideline for decision making.

Cost impact of flexibility

In terms of SCIS uncertainty and flexibility, the system infrastructure ascertains how many kind of processes can be supported with low development cost, namely in flexible-to-use category. In the parameter list - Table 1, only parameter x (process type uncertainty share) is directly related. By inspecting cost ingredients it is found that C_D is a first order function to x. Therefore the final cost C has linear relationship with x as well.

Figure 5 demonstrates the cost graph with changing x ranged from 0 to 100%. In this experiment, the system life time is set to 5 years with a hundred process types developed gradually. Each process takes 2 weeks to be developed at the unit cost of

500 or 1500 for uncertain type (FTC). Lorenz curve curvature 5 is selected to represent a medium process transaction variable level.

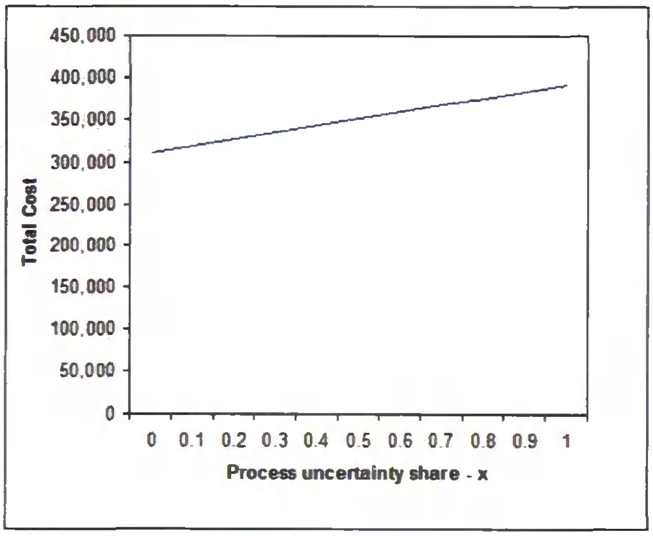
The detailed parameter assumption used in the experiment is given below:

$C_I = 20000$, $\mu_m = 5$, $\mu_p = 0.5$, $w_1 = 0.1$, $w_2 = 0.8$, $D = 20$, $v = 5$, $T_f = 10000$, $F = 100$, $dc = 500$, $dc' = 1500$, $\tau = 80$

Substitute into the cost model, we get result set in Figure 5.

Figure 5. Uncertainty share x - cost C linear graph

x	C_D	C
0	45,000	311,403
0.1	53,000	319,403
0.2	61,000	327,403
0.3	69,000	335,403
0.4	77,000	343,403
0.5	85,000	351,403
0.6	93,000	359,403
0.7	101,000	367,403
0.8	109,000	375,403
0.9	117,000	383,403
1	125,000	391,403



In model analysis presented in [6], the process uncertainty impact on the total cost is inspected on a coarse-grain level at 0.8, 0.5 and 0.2 (notice that uncertainty p in Gebauer's model has an opposite meaning with the x here). Despite the truth that cost figures in these two models are not directly comparable, the cost increasing trends are expected to be similar. However, interestingly, Gebauer's research shows a turning point in the middle ($C_{p=0.8} < C_{p=0.2} < C_{p=0.5}$) when transaction variability is normal (check costs at $v = 0.4$ for example). This result might be the inaccuracy coming from the timeless limitation in Gebauer's model [6].

In this model, unless a certain process transaction rate is too low that results in cheaper manual handling costs (set excessive w_2), it is guaranteed that higher uncertainty rate always results in more expensive overall cost since $dc < dc'$.

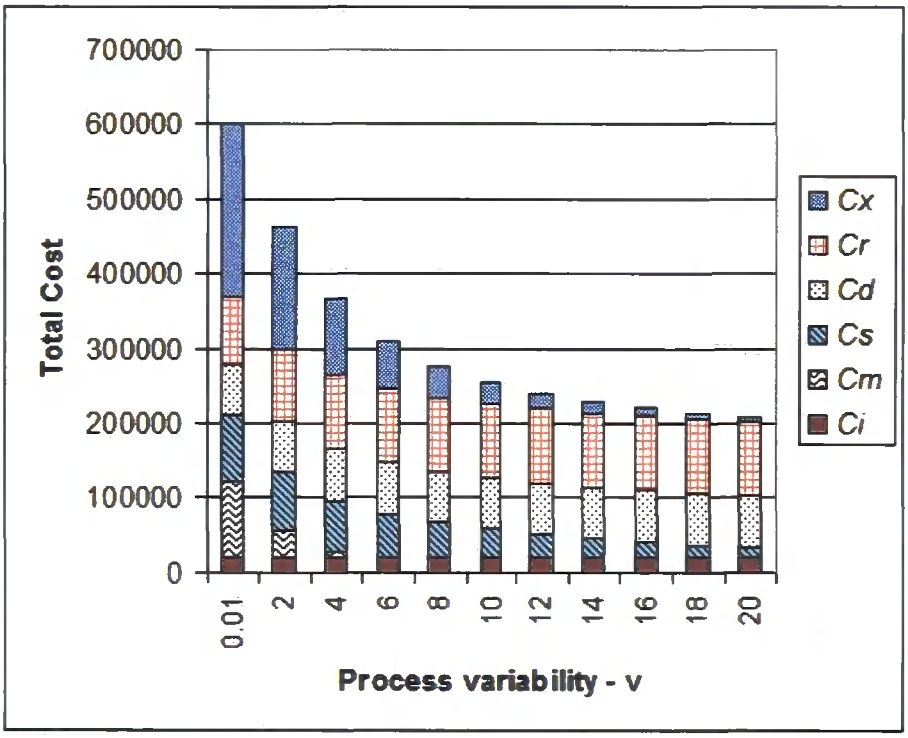
Cost impact of process variability v

Process variability is the parameter incorporated in most of the cost ingredients. Same as above approach, the model output is plotted to achieve a visual cost transition. In this test case, process uncertainty rate x is fixed to 0.3 and variability v is increased gradually within the range of $v \in [0.01, 20]$. All other parameters maintain the same values as that given in the flexibility test.

Table 2. Process variability v - cost C matrix

v	C_I	C_M	C_S	C_D	C_R	C_X	C
0.01	20,000	99,551	89,955	69,000	90,045	232,972	601,523
2	20,000	34,653	79,036	69,000	96,535	164,669	463,893
4	20,000	9,176	66,417	69,000	99,082	104,240	367,916
6	20,000	2,043	54,769	69,000	99,796	64,854	310,461
8	20,000	411	44,914	69,000	99,959	41,593	275,877
10	20,000	78	36,785	69,000	99,992	27,755	253,610
12	20,000	14	30,119	69,000	99,999	19,165	238,297
14	20,000	3	24,660	69,000	100,000	13,588	227,250
16	20,000	0	20,190	69,000	100,000	9,831	219,021
18	20,000	0	16,530	69,000	100,000	7,225	212,755
20	20,000	0	13,534	69,000	100,000	5,375	207,909

Figure 6. Process variability v - cost C graph



As Figure 6 reveals, process variability v has critical influences to C_X, C_S and C_M. The final cost C, as the sum of these ingredients, presents a downward converging trend towards $(C_I + C_D + C_R)$ along with process variability increase. At the far right end of the trend, process transactions concentrate on single process type which is developed in the very beginning (within w_1 proportion).

The cost model reveals process uncertainty and variability are irrelevant as uncertainty is about process type whereas variability is a measure on process transaction diversity. In the sense of software engineering, high uncertainty gives challenges to project planning, testing and bug control, thus results in extra overheads. Details of this kind of inter-relationship is out of the scope of this research but covered in many software engineering publications such as [9,14,15].

Process variability is a stationary character of most supply chain practices. Despite of its effectiveness to cost reduction, it remains stable roughly in real supply chain practices. To achieve a good planning towards cost effective objective, the process type share w_1 and w_2 should be of more interesting to system designers.

Issues regarding system development

In this cost model, supply chain information system affects the total cost from parameters of C_I, μ_p, w_1, w_2, dc, dc', τ. Several propositions are stated here as an information system implementation guideline. This guideline is not meant to replace software project director's working plan, rather, it targets to strategic decision makers for healthy investment and information system budget allocation.

- **Proposition 1** - Appropriate choice of software infrastructure reduces C_I, μ_p, w_1, dc, dc', τ

  A good system framework can greatly reduce further development cost and time by its well-designed business object libraries, templates and manual resources (lower dc, τ) and extensibility (lower dc'). However, its initial cost could go higher due to relative expensive price of technical supports, difficulties and training requirements (higher C_I). Moreover, good integratibility to existing production systems can reduce or even eliminate efforts to travel the data in between. This leads to lower unit process transaction cost μ_p.

  Turnkey systems, on the contrary, have relatively high w_1 and low μ_p that have positive contributions to cost effectiveness. Its penalty could also be considerable due to its sheer re-development difficulty (high dc, dc', τ).

- **Proposition 2** - Early project preparations move share w_2 into w_1, and consequently reduces final cost C

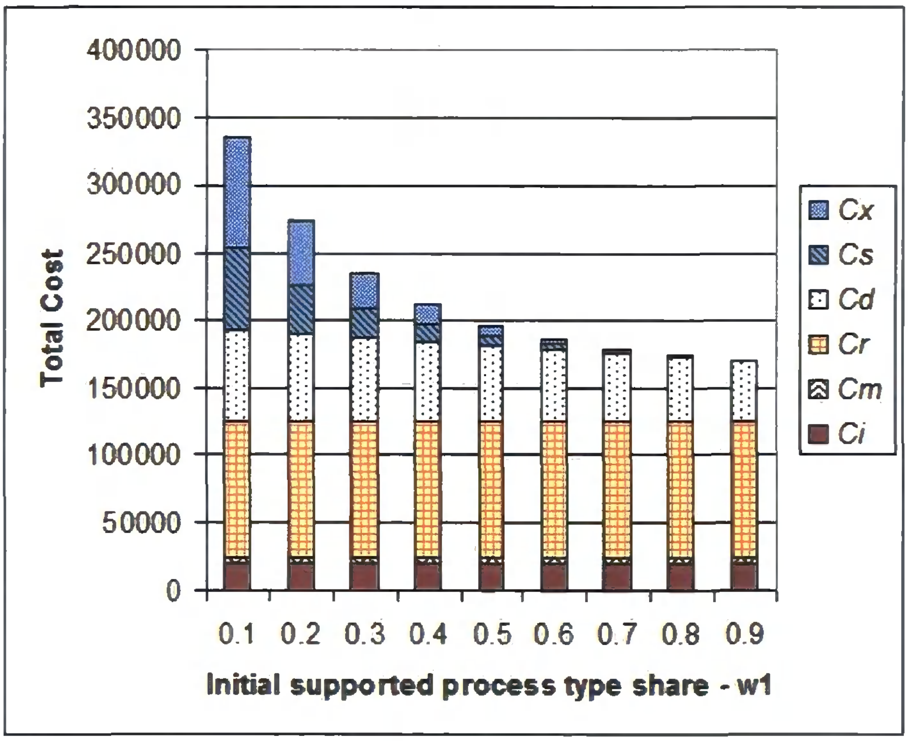
  Early and good preparations and their benefits has been recognised as "common sense" all over the areas of software engineering and even backed up by general project management [15,16,17], but why and how important it is? With the help of this cost model, Figures from a simple experiment can tell the answer: Keep manual process share constant (i.e. $w_1 + w_2 = 0.9$), let w_1 increase from 0.1 up to 0.9 to inspect the output (w_2 decreases from 0.8 to 0 accordingly). All other parameters remain the same values as used in earlier experiments.

  $C_I = 20000$, $\mu_m = 5$, $\mu_p = 0.5$, $D = 20$, $v = 5$, $T_f = 10000$, $F = 100$, $dc = 500$, $dc' = 1500$, $\tau = 80$

Table 3. Early development share w_1 - cost C matrix

w_1	w_2	C_I	C_M	C_S	C_D	C_R	C_X	C
0.1	0.8	20,000	4,401	60,386	69,000	99,560	82,056	335,403
0.2	0.7	20,000	4,401	36,359	66,000	99,560	47,390	273,710
0.3	0.6	20,000	4,401	21,786	63,000	99,560	26,681	235,428
0.4	0.5	20,000	4,401	12,947	60,000	99,560	14,438	211,345
0.5	0.4	20,000	4,401	7,586	57,000	99,560	7,328	195,875
0.6	0.3	20,000	4,401	4,334	54,000	99,560	3,333	185,628
0.7	0.2	20,000	4,401	2,362	51,000	99,560	1,227	178,549
0.8	0.1	20,000	4,401	1,166	48,000	99,560	266	173,392
0.9	0	20,000	4,401	440	45,000	99,560	0	169,401

Figure 7. Early development share w_1 - cost C graph



It is clear that early development share w_1 has major impact on C_X and C_S. The final cost C, when $w_1 = 0.9$ (namely, process types are fully implemented at the beginning apart from those 10% share left to manual handling) is halved compares to upgrade based development. As a conclusion, it is strongly suggested that the best practice is to have as much process types as possible implemented at early stage of development, when possible.

- **Proposition 3** - Sophistication and size of system developing team influence μ_p, dc, dc', τ

  It is obvious that good system design and implementation result in an efficient supply chain system (lower μ_p) and a sophisticated team spends less money and time (lower dc, dc', τ), no matter the software project is in-house developed or outsourced. One important thing need to be balanced is the size of the developing team: resource consumption (salary, management overheads, computers, workspace, etc.) of a developing team increases along with its size proportionally but not necessary the *productivity*. It is also a waste to maintain a large team on a small scale supply chain project where uncertainty is high.

  It is not the capability of this model to give size balancing indications yet important to a cost effective practice. Further study could refer to [4,15] and followings.

Model deviations and possible improvements

Presented in the research, a few simplifications are employed at various stages. These simplifications inevitably introduce deviations that worth being addressed here for future research.

Firstly, it is arguable that the approximate function of Lorenz curve in this paper has the best fit. Nevertheless, it has been made easy enough to replace the formula with another choices - the only hindrance is probably that the form of C_X would not be that straight-forward and computer based iteration becomes essential.

Secondly, the utilisation of τ introduces another major cost deviation that could be considerable in certain situations such as development team fluctuation or huge divergence of process implementation difficulty.

Finally, total cost from this model is a plain accumulation of all processes. From software project engineering it is suggested that final cost is an exponential function of the project size [15]. It is still not clear how to apply this adjustment to this model. Future study is suggested on this topic.

CONCLUSIONS

This paper presented a cost model that is dedicated to supply chain information systems where majority of development work is done by system upgrade due to its high uncertainty and dynamic environment. The model is established on a sequential development pattern which is appropriate for normal SCIS practices since resources and team productivity are usually specified in project proposal.

By model experiments, it is proved that in SCIS, total cost is a first order function to system flexibility. An appropriate SCIS infrastructure increases flexibility degree and the unit development costs, therefore has positive contribution to cost effectiveness. Process variability has major impacts on various cost ingredients and leads to a slowdown converging trend of total cost when it increases. Although process variability sounds more preferable to cost reduction, it is one of the supply chain characters that can't be varied. A few propositions are also discussed to establish the relations between model parameters and system development elements.

With this cost model, it is relatively easy to obtain the cost estimation by feeding suitable parameters into it. Benefits from the mathematical analysis, there are no empirical adjustment factors that are up to users to decide. This feature greatly relieves the difficulty and ambiguity of model application. Therefore, in most cases the model results should be close enough to the actual *minimal* SCIS cost from the estimation and budget allocation point of view.

This research investigates SCIS cost solely from the process point of view. A few limitations are resulted from the process view as well. The details have been addressed in the model deviations section for future research, to formulate a more precise yet complex cost estimation.

REFERENCES

[1] Applegate L.M.; McFarlan F.W. and McKenney J.L. (1999) Corporate Information System Management: The Challenges of Managing in an Information Age, Boston, MA:Irwin/McGraw-Hill

[2] Boehm B. (1981) Software Engineering Economics, Prentice-Hall, ISBN: 0138221227

[3] Boehm B. et al (2000) Software cost estimation with Cocomo II, Prentice-Hall, ISBN: 0130266922

[4] Brooks F.P. (1975) The Mythical Man-Month, SIGPLAN Notices, vol. 10, no. 6, pp. 193

[5] Chopral S.; Meindl P. (2004) Supply Chain Management: Strategy, Planning and Operations, Pearson Education, Inc.

[6] Gebauer J.; Schober F. (2006) Information system flexibility and the cost efficiency of business processes, Journal of The Association for Information Systems, pp. 122-147

[7] Hanseth O.; Monteiro E. and Hatling M. (1996) Developing Information Infrstructure: the tension between standardisation and flexibility, Science, Technology and Human Values, vol. 11, no. 4, pp. 407-426

[8] Lambert D.M.; James R.; Ellram L.M. (1998), Fundamentals of Logistics Management, Ch. 14, Boston, MA:Irwin/McGraw-Hill

[9] Lehman M.M. (1989) Uncertainty in computer application and its control through the engineering of software, Journal of Software Maintenance, vol. 1, no. 1, pp. 3-27

[10] Lorenz M. O. (1905) Methods of measuring the concentration of wealth, American Statistical Association, vol. 9, pp. 209-219

[11] Milgate M. (2001) Supply chain complexity and delivery performance: an international exploratory study, Supply Chain Management: An International Journal, vol. 6, no. 3, pp. 106-118

[12] Premkumar G.P. (2000) Interorganization systems and supply chain management: an information processing perspective, Information Systems Management, vol. 17, no. 3, pp. 56-69

[13] Quinn F.J. (1997) What's the buzz?, Logistics Management, vol. 36, no. 2, pp. 43-50

[14] Singpurwalla N.D.; Wilson S.P. (1999) Statistical Methods in Software Engineering: Reliability and Risk, Springer, ISBN 0387988238

[15] Sommerville I. (2000) Software Engineering 6th Edition, Addison Wesley, ISBN: 020139815X

[16] Murch R. (2000) Project Management: Best Practices for IT Professionals, Prentice Hall PTR, ISBN: 0130219142

[17] Hedeman B.; Heemst G.V.; Fredriksz H. (2005) Project Management Based on PRINCE2, Van Haren Publishing, ISBN: 9077212833