3-30-2005

A definition and measure of workflow modularity

Dawn-Marie Chin
Florida International University

DOI: 10.25148/etd.FI14060805

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

A DEFINITION AND MEASURE OF WORKFLOW MODULARITY

A thesis submitted in partial fulfillment of the

requirements for the degree of

MASTER OF SCIENCE

in

INDUSTRIAL ENGINEERING

by

Dawn-Marie Chin

2005

To: Dean Vish Prasad
College of Engineering

This thesis, written by Dawn-Marie Chin, and entitled A Definition and Measure of Workflow Modularity, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this thesis and recommend that it be approved.

Martha A. Centeno

Chin-Sheng Chen

Ronald Giachetti, Major Professor

Date of Defense: March 30, 2005

The thesis of Dawn-Marie Chin is approved.

Dean Vish Prasad
College of Engineering

Dean Douglas Wartzok
University Graduate School

Florida International University, 2005

DEDICATION

I dedicate this thesis to my husband and my mother. Thank you both for believing in me. I love you.

ACKNOWLEDGMENTS

I would like to thank my committee members, Dr. Ronald Giachetti, Dr. Martha Centeno and Dr. Chin-Sheng Chen. Thank you for serving on my committee.

Most importantly, I would like to express my sincere gratitude to Dr. Ronald Giachetti, my major professor. Thank you very much for your time and patience.

ABSTRACT OF THE THESIS

A DEFINITION AND MEASURE OF WORKFLOW MODULARITY

by

Dawn-Marie Chin

Florida International University, 2005

Miami, Florida

Professor Ronald Giachetti, Major Professor

The purpose of this study is to define and measure workflow modularity. There is an increasing need for organizations to implement processes that can be easily configured to offer distinctive capabilities compared to the competition. The concept of modularity provides the foundation for organizations to design flexible processes.

The Event-Driven Process Chain (EPC) approach is used to model an example workflow to illustrate. Based on the model of atomic tasks, rules are developed to guide the creation of modules with high cohesion between tasks in a module and loose-coupling between modules. Matrices of atomic tasks interdependencies are developed and tasks are then clustered based on interdependence strengths.

The main deliverable is a mathematical model for defining and analyzing a modular workflow to enable the creation of flexible workflow processes. The modularization model represents tasks relationships that maximizes cohesion between tasks, minimizes coupling between modules, while minimizing workflow time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

1 INTRODUCTION

The global economy is changing business practices. The failure or success of an organization is frequently based upon the organization's ability to respond quickly to changing customer demands and to utilize new technological innovations. The firm that can offer greater varieties of new products/services with higher performance and greater overall appeal will have the advantage to satisfy a complex set of customer requirements. To be competitive, organizations need to implement processes that can be easily configured to offer distinctive capabilities compared to the competition. In order to offer a wide variety of products but maintain the economies of scale that comes from large production runs many companies have started to utilize modular designs of both products and services.

Modularity provides a rational means to enhance the flexibility of existing product or process solutions. It has been adopted in a number of industries, such as automotive, computer technology and software development. Much of the research on modularity as a strategic approach has been to address four main concerns of new product development. These concerns are: 1) a structured approach to dealing with complexity, which looks at how to handle interacting and interdependent parts in complex systems, 2) responsive manufacturing through flexibility/agility, which addresses the ability to rapidly change processes in response to demand [1], 3) efficient deployment of stakeholder requirements, and 4) a rationalized introduction of new technology, which refers to a structured approach to implement new technology and satisfy customer and other stakeholder requirements .

Many enlightening studies have examined examples of modularity in product designs, such as elevator systems, using standard component interfaces ([2], *and* the automobile industry, where common components are used in many different models [3]. However literature suggests that the strongest impact of product modularity has been in the computer industry, where a family of computers of different sizes shared the same instruction set and peripherals [4]. The benefits gained in the computer industry have served as a prime example for companies to better handle product complexity by breaking up products into subsystems, usually called modules.

Product modularity has a strong relationship with the modularity of processes and resources [5]. Kusiak [5] recognized that in the analysis of a product, process or resource model for modularity the perspectives presented to the user for validation and optimization are common to all three models. Analysis of product modularity in several cases [6-9] considers the corresponding life-cycle processes for product components. However, despite all the work that's been done for product architectures, very little has been done in exploring what constitutes an appropriate modularization of workflow processes, namely tools for defining general modules. To be able to apply modularity concepts to workflow processes requires an understanding of the key concepts of modularity and workflows and how they are modeled to facilitate identification and analysis of their components.

The remainder of this chapter examines the concepts of modularity, through applications to product architectures, and the challenges and benefits gained. This is with a view to apply similar concepts to modularize workflows. The chapter also outlines issues related to processes, workflows and the coordination of dependencies between tasks and identifies aspects of modularity already implemented, and states the goals and objectives of the research. Chapter 2 provides a review of relevant literature. Chapter 3 outlines the methodology in defining and measuring modular workflows. Chapter 4 outlines the approach taken to modularize workflows. Chapter 5 detail how to measure the workflow modularity and Chapter 6 present conclusions and further work on modular workflow redesign.

1.1 PROBLEM STATEMENT

Organizations are realizing the increasing need to implement processes that can be easily configured to offer distinctive capabilities compared to the competition. The concept of modularity provide the necessary foundation for organizations to design different products/services, thereby reaping benefits of mass production, such as economy of scale, increased feasibility of product/component change, increased product variety, reduced order lead time, decouple risks [10],[11], and strategic flexibility [12]. Very little has been done in applying modularity concepts to workflow processes. This may be due in part to that fact that even though modularity has been recognized as a good design practice, there are still several issues being addressed. The main issue is a lack of formal theory and tools for defining modules from a broad perspective. Kusiak [5] stated that there is tremendous growth and potential in modularity, that can be realized by moving outside of the current practice of applying modularity solely to products industry, to redefining and enlarging the domain to include different processes and technologies, and incorporating product life-cycle cost in order to improve the quality of modules developed.

In order to define modular workflows we need to answer the following research questions:

- What are the key concepts of modularity?
- What is a process?
- What is a workflow?
- What is required to have a work process fragment, i.e. a process module?
- What makes a workflow process modular?
- How does one define process modules?
- How can one define coupling between modules?
- How does one determine potential workflow configurations?
- What effect will different configurations (flexible workflow) of modular workflow have on key performance indicator such as timing?
- How can modularity be measured?

In searching for answers to these questions, it became obvious from published literature that much of the research focused on understanding modularity concepts, modular product designs and the measurement of modularity on product architectures. There has been very little work investigating modularity concepts with respect to processes.

Table 1 lists some examples of modular products that offer a large number of product variations. Here we see the use of modularity to create loosely coupled component designs through the standardization of component interface specifications, and product architectures composed of the relationships between components. These relationships are defined by the specifications of inputs and outputs linking each component in a design and component interface specifications. Loosely coupled components describes components that enable substitution into different product designs, without requiring redesign of other components [12]. The standardization of interfaces refers to loosely coupled components that can be treated as a 'black box' [13].

Table 1: Examples of Products with Modular Designs

References	**Product**	**Form of Modular Design**
Langlois and Robertson [14]	Personal Computers	Personal computers consist of modular components such as hard disk drives, flat screen displays, and memory chips coupled with a microprocessor chip and enclosure.
Nevins and Whitney [15], Tully [16]	Automobiles	Modular components for different models
Sanderson and Uzumeri [17]	Consumer electronics	'Mixing and matching' modular components in a few basic designs for over 160 variations of Sony

References	**Product**	**Form of Modular Design**
		Walkman
Sanchez and Sudharshan [18]	Household electronics	General electric uses different modular doors and controls on common assemblies of enclosures, motors, and wiring harnesses on several modules of dishwashers.
Cusumano [19]	Software	Modules of routines, which can be combined to create customized applications programs.
Woolsey [20]	Aircraft	Common wing, nose, and tail components allow several models to be leveraged by using different numbers of fuselage modules, creating aircrafts of different lengths and capacities.

Much of the literature focuses on modularity measures, which focus on shared relationships between components, respective interfaces, standard components, substitutability of components and the impact of design alternatives. The literature revealed that studies range from exploratory qualitative measures to quantitative measures that apply optimization models to address manufacturing issues. Table 3 in section 2.1.1 on page 26 details the various measures explored.

Some of these measures can be adapted for modularizing workflow processes, where the similar issues regarding decomposability (i.e. modularization) and integration of modules are being considered. Organizations are required to understand shared module relationships, interfaces, and standardization of modules and substitutability of modules across organization in to create processes that can be reconfigured easily and quickly based on competition. The measurement approach presented by Gershenson [6] is explored in measuring modularity in workflows, as it focused on similarities between modules and their life-cycle processes and dependencies that exist between the modules and processes.

Sanchez and Mahoney [12] used the concepts of nearly decomposable systems[1] to investigate concepts of modularity in product and organizational designs. Their approach examined the outputs of component processes that are partitioned into tasks that can be operated autonomously and concurrently, which enabled a new approach to knowledge management in modular organization designs. They concluded that embedded coordination of standardized interfaces of modular architectures is attainable in other

[1] A nearly decomposable system is one in which interactions among sub-systems are weak.

processes and may be a new design approach for achieving increased flexibility and inter-organizational connectivity among broadly de-integrating (loosely coupled) organizations.

Therefore, similar to product designs, modular process designs can be defined by specifying process modules[2] with common activities that can be combined to create customized process workflows. These modules will be loosely coupled to enable substitution across organizations without requiring redesign, through the use of standard module interface specifications.

A workflow is a process in an enterprise, which is coordinated by software called a Workflow Management System [21]. Workflow management systems control the flow of work and information, through the use of standardization mechanisms. Today, workflow management systems provide a solution approach to rising issues including real-time collaboration, information and flexibility, personalization (mass customization), quality and function versus process. However, so far conventional workflow software only covers individual functions, such as procurement or accounting, and not much focus has been placed on the interaction of these functions within a process.

The problem of modularizing workflow processes requires determining an appropriate measure of process modularity, defining the number of possible configurations for a process and creating a model for analyzing these configurations.

[2] Note that process modules hereby used in this paper to refer to modules in a process design performing a function within a system of interrelated modules whose collective functioning make up the process and whose relationships are defined by the inputs and outputs linking modules in a design.

1.2 GOALS AND SPECIFIC OBJECTIVES

The main goal of this research is to study the application of modularity concepts to workflow processes. To accomplish this goal, the examples of approaches used in modular product designs are adopted.

To achieve this goal, the specific objectives include:

1. Defining a modular workflow process,
2. Developing a mathematical model for defining modules of a modular workflow process, including identifying potential configurations
3. Developing a measure for workflow process modularity,

Details on how each of these objectives is achieved are given in Chapter 2.

CHAPTER 2

2 LITERATURE REVIEW

This chapter discusses research conducted in modularity and workflow management. Section 2.1 focus on the key concepts of modularity through an examination of work that has been done for modular product architectures, then outlines modularity measures that have been explored, and finally details the modularity measure presented by Gershenson [6]. Section 2.2 focuses on workflow processes, and Section 2.3 explores the application of modularity concepts to workflow processes, including examples of commercial workflow process systems. This review helps in understanding the key concepts of modularity, processes, workflows and how they are modeled to facilitate identification and analysis of their components.

2.1 MODULARITY

To handle complexity of large systems humans have learned to divide them into smaller pieces and study each piece separately. Modularity is a concept that has been used in various fields to divide these complex systems so they can be more easily designed and managed. Baldwin and Clark [4] identified five key areas in the general concept of modularity, including:

1. Interdependence within modules, which refers to the strong connections between structural elements within module units
2. Independence across modules, which refers to the relatively weak connections to elements in other units

3. Abstraction, which enables the hiding of complexity of elements in a complex system, by breaking-up the system into smaller simplified pieces
4. Information hiding, refers to design parameters that are hidden from the rest of the system, and
5. Interfaces, which are descriptions on how the different modules interact. They are the visible information, which constitutes the design rules.

Kamrani and Salhieh [22] referred to a sixth key area of modularity:

6. Design standardization, where standard components are shared.

Baldwin and Clark [4] also purported that a set of design rules must include categories of design information, including a architecture of what modules will be part of the system and their roles, interfaces on how the modules interact, and integration protocols and testing standards that allow for designers to determine how well a system works.

In exploring the concept of modularity many studies have been done on modular product designs and the advantages gained from the use of standard components. The remainder of this section will organize the studies based on the key areas of modularity identified by Baldwin and Clark [4] and Kamrani and Salhieh [22].

Baldwin and Clarke [23] further studied modularity and how to manage in an age that is fast adopting modular design concepts. They defined modularity as an efficient way to build a complex design from smaller subsystems that can be designed independently yet function together as a whole. They further expanded on a guide to modularity, through the partitioning of information into visible information and hidden information, thus specifying which parameters will interact outside of modules and how these interactions across modules will be handled.

Kamrani and Salhieh [22] studied modularity in product designs. They defined modularity to be the process of producing units that perform discrete functions, that when connected together provide a variety of functions. They identified that modular designs should focus on minimizing interactions between components, which enable independent design and production of these components. They also noted that product modularity can be represented in several ways and are based on the types of combinations between modules, which are determined by the type interactions between different modules in the product. These include component-swapping modularity[3], component-sharing modularity[4], fabricate-to-fit modularity[5] and bus modularity[6].

Yigit et al. [24] looked at the problem of optimizing modular products in reconfigurable manufacturing system. They addressed the issue of determining optimal number of module instances and selecting the optimum subset of module instances from a large number of alternatives. The problem is first posed as a subset selection problem, and then transformed into an integer nonlinear programming problem. The methodology

[3] Here different product variants of the same product family are created by combining two or more alternative types of components with same basic component or product (Kamrani and Salhieh, 2000).

[4] Different product variants of different product families are created by combining modules sharing the same basic component (Kamrani and Salhieh, 2000).

[5] Here one or more standard components are used with one or more infinitely variable additional components (Kamrani and Salhieh, 2000).

[6] A module can be matched with any number of basic components. This allows the number and location in a product to vary (Kamrani and Salhieh, 2000).

was proven to be a good tool for selecting module instances and for designing modular products.

Kamrani and Gonzalez [25] proposed a genetic algorithm-based solution methodology for modular design. They viewed modular design as the process of producing units that perform discrete functions, which are then connected together to provide a variety of functions. The modular design problem was formulated as set of combinatorial optimization problem using the design for modularity methodology introduced. A genetic algorithm (GA) and heuristic-based GA were proposed to solve the problem.

Ethiraj and Levinthal [8] looked at modularity and innovation in complex systems. They review modularity concepts and their application to addressing the problem of designing, coordinating and managing complex systems. They defined modularity as a general set of design principles for managing the complexity of large-scale interdependent systems and highlighted two areas in which modularity can be deployed. These included the 'real' underlying structure for a given design problem, giving the partitioning and decomposition of tasks and the interfaces among design elements, the 'appropriate' number of modules. They proposed models based on these areas and evaluated performance using a simulation model.

Later Gershenson, et al. [26] reviewed the research on measures and design methods for product modularity. They concluded that from all the approaches, due to a lack of understanding of what modularity is, there is a general lack of consensus on modularity measures and modular product design methods. They highlighted the few areas of include the overall structure of modularity measure, normalization of the measure and measures of independence and similarity. Conflicting areas include implementation of design, role of the measure in the design method and impact on the inclusion of multiple life-cycle stages.

Kusiak [5] reviewed product and process design with a modularity perspective. He identified three key areas of unrealized potential and growth of modularity, namely, product variety, technology variety and time. He also purported that the narrowness of the domain is the main criticism of modularity, including lack of tools to define general modules. He reviewed the approaches proposed to solve this problem, including the mathematical programming notion, the biology-based notion, modularity algorithm, and model evaluation. He also reviewed the IDEF methodology as a tool for process modeling.

Mikkola and Gussmann [2] examined managing modularity of product architectures. They defined modularity as a new product development strategy in which interfaces (linkages) are shared among components in a given product architecture and are specified and standardized to allow for greater substitutability (sharing) of components across product families. They introduced a modularization function for analyzing the degree of modularity in a given product architecture, considering components, degree of coupling and substitutability. The function was applied to traction and hydraulic elevator systems. The analysis captured the sensitivity and dynamics of the systems created by three types of components (standard, neutral, and unique) and two types of interfaces (fundamental and optional).

Standardization was also identified as a key area of modularity, namely in specification of component relationships [12]. It can be achieved through the identification of component functions and the minimization of interaction between a component and the rest of the product [22].

Thormann and Brandeau [27] looked at approaches to determine the optimal level of component commonality for end-product components that do not differentiate models form customer's perspective. A mathematical program considering production, inventory holding, setup and complexity costs were proposed for wire-harness design problem. To solve small and medium-sized problems a branch-and-bound algorithm was used to find optimal solutions and a simulated annealing algorithm used to find good solutions to large-size problems. Both algorithms were applied to a wire-harness design. The optimal solution showed a reduction in component variety and costs savings compared to the no-commonality solution.

Worren et al. [28] explored modularity and its application to the home appliance industry. They explored the premise that modular product architectures and process architectures are prerequisites for efficient mass customization and cycle time reduction, thus sources of increased strategic flexibility. Based on a conceptual model linking market context, strategic flexibility, innovation orientation and organizational architecture to firm performance, they sought to prove three propositions and two research questions. Through the use of technology management and organizational and strategic management theories, they concluded that model variety was related to firm performance, product modularity was related to model variety, innovation climate was correlated with modular processes, and codification and standardization are necessary prerequisites for achieving high levels of process flexibility. There was a complex relationship between managerial cognition, market context and the use of modular architectures, and there was no significant relation between product modularity and two indicators of strategic flexibility, new model introduction and new product introduction.

Table 2 below summarizes the key concepts of product modularity and outlines similar concepts that can be adapted for modular process designs:

Table 2: Product Modularity vs. Process Modularity

Product Modularity	**Process Modularity**
Shared interfaces among product modules [2, 12]	Shared interfaces among process modules
Standard interfaces [4, 27, 28]	Standard interface specifications
Substitutability of components [4]	Substitutable process modules
Product Architecture for arranging functional elements of the product [4]	Process model for decomposing key activities, with interactions within modules
Loosely coupled components [4, 8, 12, 24, 25]	Loosely coupled modules

In summary, the design of modular process will require that the modules of a process be specified based on the following key concepts:

- Modules must have their main functional interactions within rather than between modules,
- Modules consist of autonomous (independent) tasks, which refer to the reduction of task interdependencies, through the specification of standard interfaces.

- Modules consists of concurrent (parallel) process units that can operate independently, performing discrete functions and can be tested in isolation from the system.
- Modules are loosely coupled, enabling different independent modules to be substituted (shared) across organizations process designs without requiring a redesign of other modules.

2.1.1 Modularity Measurements

Based on the literature review of modularity measurements, the approaches seemed to focus on shared relationships between components, respective interfaces, standard components, substitutability of components and the impact of design alternatives. The studies ranged from exploratory qualitative measures to quantitative measures that apply optimization models to address manufacturing issues.

Table 3 outlines approaches that have been used to measure modularity in product architectures.

Table 3: Measurements of Modularity

References	**Purpose**	**Measure**
Mikkola and Gassmann [2]	Analyzing the degree of modularity in a given product architecture, considering variables such as components, degree of coupling, and	Modularization function for analyzing the degree of modularity

References	**Purpose**	**Measure**
	substitutability of components	
Ali and Gonzalez [29]	Developing similar components in designing complex products	Genetic algorithm-based method used in the solution of a set of combinatorial optimization problems
Fisher, et al. [30]	Examining variations in component sharing and identifying factors to explain variation	Mathematical model, complemented with optimization, simulation, and regression analysis
Gershenson et al. [6]	Measuring the degree of modularity of a product based on intra-module and inter-module similarities and dependencies compared to all similarities and dependencies.	Mathematical model for measuring relative modularity based on a modularity evaluation matrix of component dependencies and similarities
Ulrich & Ellison [31]	Developing a theory for determining when a firm can benefit from product-	Regression analysis based on surveys of products.

References	Purpose	Measure
	specific component designs	
Newcomb et al. [9]	Measuring modularity based on the multiplication of inter-module connections and the average correspondence between modules	A Mathematical model that is based on a product decomposition and module comparison approach
Ulrich et al. [32]	Estimating impact of designing alternatives on the economic benefit of a product	Economic model to illustrate the relationships among DFM, lead time and profits
Evans [33]	Illustrating the application of standardization as an optimizing procedure	A non-linear programming model

The following paragraphs focus on the approach presented by Gershenson et al. [6], as this is the measure that has been adopted to measure modularity in a workflow process. The measure is chosen because it focuses on creating modules that encourage independence between components and all life-cycle processes in different modules, and similarity within components and processes in a module. The authors presented a mathematical model that measures the degree of modularity of a product based on intra-module and inter-module similarities and dependencies compared to all similarities and dependencies between life-cycle process and components. The measure focuses on shared relationships between components, respective interfaces, standard components, substitutability of components and the impact of design alternatives.

Based on a breakdown of the product into it's components (using a component tree) and process graphs outlining the life-cycle processes for each component, similarity and dependency evaluation matrices were developed to compare the task and subtask of each process with each component. The relationships were collected in an *n x n* matrix, where the first value of each cell represents similarity weights and second value represents dependency weights. Similarity referred to the same manner of processing components in a module [7] and dependency relates to the component interactions arising from the various processes the component undergoes. Using the weights of similarity and dependency relationships, six possible relationships were evaluated, Component-component dependency, component-component similarity, component-process dependency, component-process similarity, process-process dependency and process-process similarity. The evaluation of matrices considered the modularity facets of attribute independence, where component attributes had fewer dependencies on attributes in other modules; and process independence, where each task of each life-cycle process of each component in a module had few dependencies on the process of external components. These facets enabled the design of the product with increased independence and similarity. The Relative Modularity (RM) was determined as follows:

$$\text{Modularity} = S_{in}/(S_{in}+S_{out}) + D_{in}/(D_{in}+D_{out})$$

Where:

S_{in} = Similarity within a module = $\sum_{m=1}^{M}\sum_{i=r}^{s-1}\sum_{j=i+1}^{s}\sum_{k=1}^{T}\sqrt{S_{ik} * S_{jk}}$

Where, m = number of modules in the product

r = first component in the module m and module n

s = last component in the module m and module n

T = number of processes under consideration

S_{ik} is similarity between component i and task k

S_{jk} is similarity between component j and task k

The value of S_{in} is a root mean square of the similarities between the two components and a life-cycle process. S_{in} allowed component process relationships to be measured and has a positive effect as it involves grouping components with similar life cycles.

S_{out}: Similarities between the components of a module and each component external to the module

$$= \sum_{m=1}^{M}\sum_{i=r}^{s-1}\sum_{n=m+1}^{M}\sum_{j=r}^{s}\sum_{k=1}^{T}\sqrt{S_{ik} * S_{jk}}$$

Where, i,j are components not in the same module, and n is a module.

The value is based on the ratings of the component process similarity interactions for each component outside a module. S_{out} has a negative effect on the modularity measure as the approach reduces process similarities between components.

D_{in}: Dependencies between each component within a particular module = Component-Process interactions + Component-Component interactions

$$= \sum_{m=1}^{M}\sum_{i=r}^{s-1}\sum_{j=i+1}^{s}\sum_{k=1}^{T}(\sqrt{D_{ik} * D_{jk}} + D_{ij})$$

Where: i, j are components in the same module

D_{ik} is the dependence between component i and task k

D_{jk} is the dependence between component j and task k

D_{ij} is the dependence between component i and component j

The value is based on the ratings of the component-component and component-process dependence interactions for each component within a module. D_{in} has a positive effect on the measure as its important to group dependent component.

D_{out}: Dependencies between the components of a module and each of the components that are external to the module = Component-Process interactions + Component-Component interactions

$$= \sum_{m=1}^{M}\sum_{i=r}^{s-1}\sum_{n=m+1}^{M}\sum_{j=r}^{s}\sum_{k=1}^{T}(\sqrt{D_{ik} * D_{jk}} + D_{ij}$$

Where: i, j are components not in the same module

D_{ik} is the dependence between component i and task k

D_{jk} is the dependence between component j and task k

D_{ij} is the dependence between component i and component j

The value is based on the ratings of the component-component and component-process dependence interactions for each component outside a module. D_{out} has a negative effect on the total measure as external dependencies are minimized for independent process modules.

A mechanical pencil, figure 1, was used to illustrate the measure.

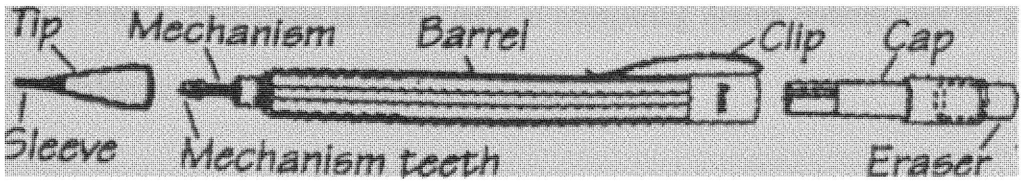


Figure 1: View of a Mechanical Pencil highlighting the cone/tip, clutch/teeth, barrel and eraser modules

The relative modularity was calculated from an evaluation of the matrix of component interactions (similarity and independence) between all the components of the pencil and the tasks involved in each process for each component. A 5-point weighting scale for both similarity and dependency was used. This was illustrated for the cone/tip assembly of the pencil considering only its components, function and manufacturing interactions as follows:

S_{in}, S_{out}, D_{in} and D_{out} were calculated using the above formulas for each of the components. Once the calculations for S_{in}, S_{out}, D_{in} and D_{out} are done for all the four modules of the mechanical pencil, the relative measure for the product was calculated by summing the relative measures of each module, as shown below:

Module	S_{in}	S_{out}	D_{in}	D_{out}	**RM**
Cone/Tip	0	80	15	98	0.13
Teeth/Clutch	45	75	131	101	0.94
Barrel	0	20	5	39	0.11
Eraser	5	35	20	40	0.46
Total Relative Modularity					**0.87**

The pencil scored low with a relative modularity of 0.87, where the possible range of values is 0 to 2. This value is useful only in comparing design options and for guiding process redesign.

2.2 WORKFLOW PROCESSES

In addition to managing organizational functions such as purchasing, manufacturing, marketing and sales, accounting, and human resources, many organizations are finding they also need to manage processes, such as inventory management, receiving, product development etc, which spans the functional units. It is these processes that necessitate the definition and creation of workflows.

In examining a process and specifically a business process, Hammer and Champy [34] defined a process as the collection of activities that create an output based on one or more inputs. Basu and Blanning [35] defined a process as a set of tasks that connects one set of information elements, the source, to another set of information elements, the target. All the inputs for any task must be either in the source or in the output of some other task in the process. Reijers [36] defined a business process as a collection of activities that takes one or more kinds of input and creates an output that is of value to the customer.

For the purpose of this thesis, a process will be defined as the network of activities and their relationships defined by objects/data elements that constitutes inputs to the process, which are transformed to outputs of the process.

In examining workflows two types of workflows, cased-based workflows (routine), which are more or less standardized workflows, which can satisfy a single instance of a customer's request, such as a customer order production processing; and ad-hoc workflows (non-routine), which arise from specific, temporary needs of unique project teams and are initiated when the customer's request is too specific to be worked on by standardized case-based workflows.

Schal [37] defined a workflow as a unit of work generating products and services, which are related to, or result in, customer satisfaction. He mentioned that workflows have sequential and parallel steps that involve the movement of people, documents, products and information. A workflow can be seen as the sequence and interrelation of information, activities and communications within a process.

Later, The Center for Technology in Government [38] defined a workflow as the movement of documents and tasks through a business process. A workflow can be a sequential progression of work activities or a complex set of processes each taking place concurrently, impacting each other according to a set of rules, routes, and roles. The process-modeling techniques for defining the detailed routing and processing requirements of a typical workflow were reviewed. The Center also reviewed workflow management systems and how they are used in organizations to define and control various activities in a business process.

The Workflow Management Coalition [39] based on ongoing research they have conducted from 1993 through 1995, presented a workflow reference model that provided a framework for the development of workflows and workflow management systems. They defined a workflow as the automation of procedures where documents, information and tasks are passed between participants based on defined rules to achieve business goals. The model illustrated five major components and interfaces in the workflow architecture, including a process definition, workflow interoperability, invoked applications, workflow client applications, and administration and monitoring.

Basu and Blanning [35] studied workflow analysis and the adaptation of processes to specific circumstances. They defined a workflow as a specific collection of tasks, resources and information elements in each circumstance, and as an instantiation of a process. The information element being an atomic data item or data structure; tasks are collection of information elements, having an input and an output; and resources being the entity associated with one or more tasks, which must be available if the tasks are to be executed. A meta-graph was proposed as means of representing workflows to enable more effective design of organizational processes. The approach proved to be more advantageous to other process modeling methods such as Petri-nets and UML state charts, in that meta-graphs are able to give a single representational construct for system components such as data, models and rules, and they support multiple perspectives.

Reijers [36] referred to a workflow as a business process that delivers services. It is the control dimension of a business process, which is the dependency among tasks that must be respected during the execution of the business process. He also identified four basic components of workflows that are suitable for modeling in the context of business process management, including a case component, describing each case that exist and how they are created. This component answers the what of the basic process questions; a routing component, which determines how cases are routed through the workflow, thereby answering the how; an allocation component, which specifies the classes of resources and which will take care of which work items, answering the by whom question; and an execution component, which determines when resources will actually execute the work allocated to them, answering the when of the process questions.

For the purpose of this study a workflow is understood to refer to specific instance (case) of a larger business process having its own activities, work items, resources and objects/data elements defining its inputs and outputs that can be automated for efficient progression of work to satisfy customer requirements. In our definition, we highlight computer automation of the workflow as enables controlled efficient routing and sequencing of tasks.

To manage the complexity and broad ranging links and interactions in workflows, workflow management systems are used. Workflow management systems (WFM) enable the standardization and automation of workflows. They allow the complete definition, management and execution of workflows through the execution of software whose order of execution is driven by a computer representation of the workflow logic. Many workflow management systems exhibit common characteristics, which allows for integration and interoperability between different products. Applications for workflows are divided into four categories, production workflow systems, messaging-based workflow systems, web-based workflow systems and suite-based workflow systems. Examples of commercial workflow products in each category include, *Work Center, Lotus Forms, Group Wise Web Access, and Microsoft Office Exchange.*

2.3 MODULAR WORKFLOW PROCESSES

Based on the literature review on the concepts of modularity and a workflow process, it can be concluded that modular workflow processes can be achieved by specifying the modules of a process architecture based on the following key characteristics:

- A process architecture, defining the building blocks of the process and the interactions between functional elements, that enables the definition of autonomous process modules that can be operated concurrently,
- Modules with functional interactions within the modules rather than between modules,
- Specifications of standard interfaces, defined by substitutable data elements that can be shared between process modules and that defines how the modules connect and communicate, and
- Loosely coupled process modules that will enable substitutability across organizations with similar process designs.

Hence the modular workflow process will be obtained by partitioning and standardizing the information into a process architecture specifying the modules that will be part of the process, interfaces detailing how the modules will fit, connect and communicate (visible information) and standards for measuring the module's performance.

Modularizing a workflow process presents similar problems as in modularizing any system, including difficulty finding the most suitable set of sub-modules (decomposition), and the difficulty of combining the separate sub-modules into an overall solution (integration). Several approaches have been presented as to how to handle these problems in modular designs.

This remaining section will explore literature on these concepts for modular workflow process designs and how they can be developed.

Sanchez and Mahoney [12] defined modular process architectures as decompositions of the company's key activities into specific routines and interfaces that allow frequent reconfiguration of processes, in the same way as for modular components in physical products. That is, the definition, analysis and splitting apart of key activities and specifying standardized component interfaces (relationships), that will allow for loose coupling of modules. A process decomposition method that will enable process simplification and well as concurrent processes, must allow for activities in the process design to be modeled in a generic manner independent of the specific product being designed. This method must recognize a product flow perspective, identifying inputs and outputs; an information flow perspective, analyzing precedence constraints between the design activities, thereby identifying information needed; and a resource perspective, identifying external and internal resource constraints needed to transform inputs to outputs (Kamrani and Salhieh, [23]).

Standard interfaces can be defined by examining the process module. This is the complete unit of work that has an input and output and is supported by one or more resources. The interfaces (relationships) between the inputs, outputs and resources will be defined by the parameters of the process modules within the process architecture, including the objects/data elements, tasks/activities, and resources. That is, a set of data structures defined to contain all the inputs and outputs of the set of tasks/activities of the workflow. These interfaces will describe how the modules will fit together, connect and communicate [23].

The Workflow Management Coalition (WMC) [39] defined five core interfaces of a workflow, including a Workflow Definition Interface[7], a Workflow Process Instance Interface[8], a Workflow Activity Instance Interface[9] and a Workflow Item Interface[10]. A process definition is realized by a process instance, which contains activity instances that are realized by work items. Figure 2 illustrates how a business process can be decomposed to define these base workflow interfaces:

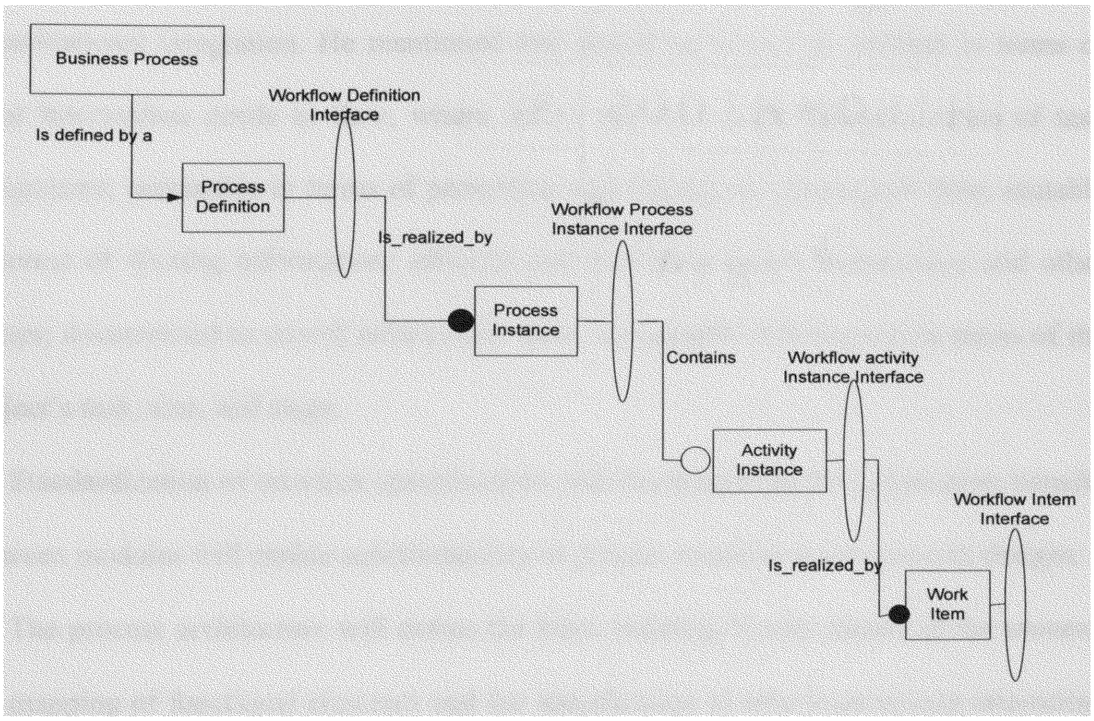


Figure 2: WMC Workflow Interfaces

[7] WMC (1997) defines as the template of a workflow process model that holds the instance of the executing process, including the set of activities and their relationships.

[8] WMC (1997) defines as an instance of a particular process definition.

[9] WMC (1997) defines as the representation of a single enactment of process (process step) in a workflow model being executed and is associated with a process instance.

[10] WMC (1997) defines as a representation of task to be performed in the context of activity within a workflow process. It provides operations to access and modify attributes of objects.

Each interface is defined by data elements that are associated with a module of the workflow process. The process instance is defined by objects/data elements of parameters of all inputs, outputs and resources. Each process instance has different activity instances of steps in the process that will require one or more resources or work items to perform activities.

Browning [40] defined characteristics of information transfer interfaces necessary for organizational integration. He mentioned that interfaces should be defined in terms of what information needs to flow, where, when and how; tight-fitting in terms of task assignment; permeable in terms of permitting and regulating information flow; mutable in terms of altering information; efficient and free from undue bureaucracy and other delays; documented to record information flow; measurable and adapted in terms of the project's task, size, and stage.

Standardization of interface specifications and characteristics for information transfer between modules will enable substitutability of process modules across process designs.

The process architecture will define the basic building blocks (tasks) of the process, the mapping of functional elements and the specification of interfaces among interacting modules. Each process module can be modeled as a domain, which defines the subset of process to be modeled, with boundaries/relationships defining the interaction with another domain through the exchange of events and results. These events and results will be represented as physical or information objects (data structures) defining the module interfaces.

Process models are used to collect and organize the data and knowledge about the processes and in effect can be used to define process architectures. Davenport [14] defined a process model as containing a set of activities arranged in a specific order, with clearly identified inputs and outputs. A standard process module can be developed based on the process models. To model these processes several modeling methodologies have been used:

Peterson [15] developed Petri Nets, a graphical process-modeling tool. It represents a process as a network of places, transitions and arcs, and tokens to represent the state of the system. The European Committee for Standardization [16] introduced Computer Integrated Manufacturing-Open Systems Architecture (CIMOSA), a reference architecture capturing both process functionality and process behavior. It decomposed the process into function, information, resource and organization views.

Since in this study we are interested in the modeling and design of modular workflows, that is enabling rapidly changing processes, producing different processes without major redesign, and the ability to offer a wide variety of product/services to customers, we will adopt the techniques of Event-driven process chain (EPC) to define functions in a workflow process. The EPC model is explained in Section 3.3.

Modularity can therefore be achieved through the decomposition of the workflow process into individual modules defined by the workflow interfaces, where each module will include inputs, outputs, controls and mechanism and specifications for all interfaces.

Kamrani and Salhieh [22] stated that integration of the several functional elements in a modular system must focus on similarities between the physical and functional architecture of the design, and minimizing or eliminating the degree of interaction (interfaces) between physical elements. Similarly for modular process designs, integration of the different modules will be based on similar criteria and the weakness of the interfaces between modules will be a measure of the strength of the design.

CHAPTER 3

3 RESEARCH METHODOLOGY

This chapter starts by defining key concepts of modularity and workflow processes, and then reviews the research methodology, a modeling approach is then presented, and then an example application that will be used throughout the next sections is outlined. The chapter is organized to focus on the steps to define workflow modularity and to measure modularity between components, with a view to achieving the following objectives:

1. Definition of general rules for defining a modular workflow process
2. Analysis of an example workflow process that is used for illustration
3. Development of matrices to record task interdependencies
4. Development of clustering model for groupings tasks into modules
5. Measurement of a workflow modularity using adopted measure

3.1 DEFINITIONS

Definition 1: Modularity

Modularity is a concept that enables process flexibility and interchangeability through the sharing of interfaces among autonomous process components. Modularity requires strong cohesion (interdependence/similarity) between components within modules and loose coupling (independence) across modules [4, 12].

Interdependence refers to the strength of the relationships between module components, whereas independence refers to the weakness of the relationships that exists between one module and another [4, 22].

Concurrent, flexible processes created in design modularization where independent block of tasks with interconnected, independent and hierarchical elements is the foundation for this work.

Definition 2: Process

A process will be defined as the network of activities and their relationships defined by objects/data elements that constitutes inputs to the process, which are transformed to outputs of the process.

Definition 3: Workflow process

A workflow is understood to refer to specific instance (case) of a larger business process having its own activities, work items, resources and objects/data elements defining its inputs and outputs that can be automated for efficient progression of work to satisfy customer requirements [35, 36]. Each workflow consists of a set of tasks/actions (or atomic work unit) specifying the work to be accomplished. Figure 3 & 4 below illustrates the definition of a workflow. Figure 3 shows the breakdown of a business process, the set of linked activities, which can be manual or automated and each activity can be further divided into individual tasks.

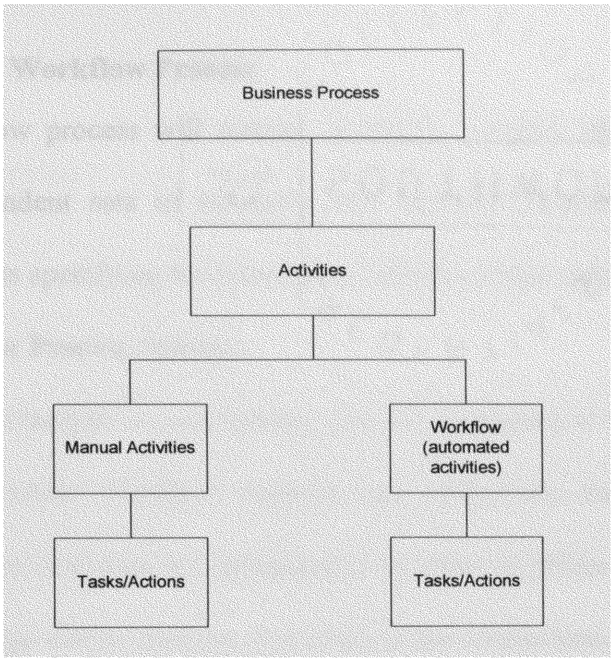


Figure 3: Breakdown of Business Process

Figures 3 and 4 illustrate the fact that a workflow consists of one or more tasks, which are realized by one or more work items.

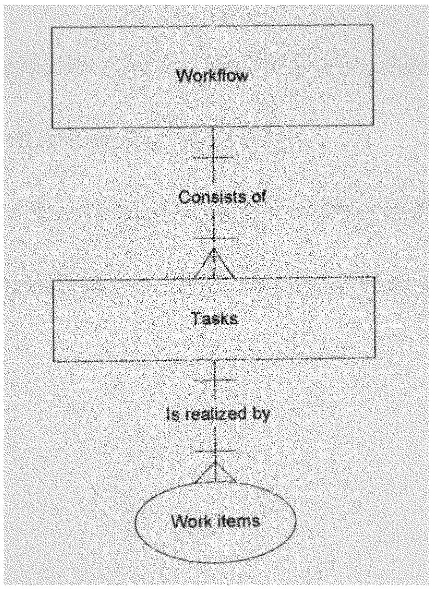


Figure 4: Workflow Definition

Definition 4: Modular Workflow Process

A modular workflow process will consist of loosely coupled, blocks of task with interconnected, independent sets of resources and standard interfaces specifications, defined by data elements specifying the interactions between tasks within blocks.

Definition 5: Workflow Process Module

A workflow process module is a component activity, consisting of interrelated atomic tasks units, within a system of related modules that collectively make up a business process. The component activities are independent of other activities in other modules and there exists couplings within modules as a result of the interactions between tasks.

Definition 5: Process Function vs. Workflow Module Function

A process function refers to the functions/roles in the process that converts inputs into outputs.

A workflow module function refers to the specific functions the module performs within the process, for example the role of the 'invoicing module' in an order fulfillment process is to process approved orders for fulfillment.

Based on the definition of the modular workflow process, the research seeks to define a methodology for designing modular workflows and a modularity measure.

3.2 DESIGNING MODULAR WORKFLOWS

In order to define modular workflows, it is necessary to study the interdependencies between tasks to enable a better understanding of the task interactions how they can efficiently grouped to maximize process objectives. A survey method is used, where questionnaires are administered to people in the organization that are knowledgeable of the process. This information is then used to develop process models for process. The Event-Driven Process Chain (EPC), a business modeling technique is used, which is explained in the next section.

Based on the process model, design structure matrices (DSM) are created based on specified guidelines to evaluate the strength of the tasks interdependencies. In the matrix row and columns headings would correspond to the workflow tasks and the cells will be dependence weights based on relationships between tasks.

A clustering algorithm can be used to group tasks into modules based on the workflow modularity objectives and guidelines. Once clustering is complete, modules can then be measured for relative modularity. Figure 5 below outlines the design approach to be followed to modularize a workflow.

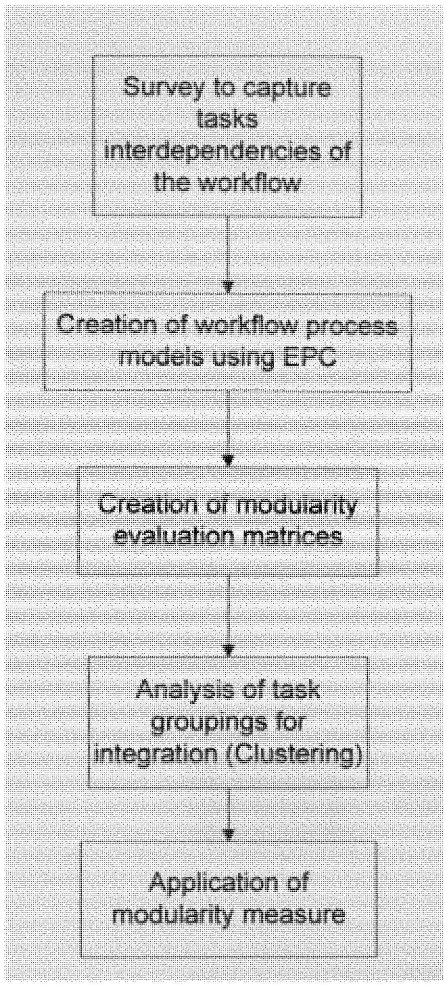

Survey to capture tasks interdependencies of the workflow
Creation of workflow process models using EPC
Creation of modularity evaluation matrices
Analysis of task groupings for integration (Clustering)
Application of modularity measure

Figure 5: Modular workflow Design Methodology

3.3 WORKFLOW MODELING WITH EVENT-DRIVEN PROCESS CHAIN (EPC)

The modeling will be based on the definition of a workflow process obtained from the survey of the process and will consider key aspects such as modeling the detailed routing and processing requirements of the workflow, thereby capturing the operation and information perspectives of the workflow. The operation perspective describes for each part of the workflow, the supporting operations, whereas the information perspective describes data consumed and produced by the workflow.

Therefore in defining the modularity of the workflow process, the workflow process structure will be examined based on the basic framework for workflow definition as defined in Section 3.1

This work supports the approach of modeling and measuring interdependencies between tasks in a workflow [41]. As such, all analysis is done at the task level of abstraction in order to capture the atomic work unit (tasks) of a workflow. EPC enable the representation of the elements of the information and operation (function) views of a workflow. This includes objects to represent events, functions, resources and information and control flows, as shown in Figure 6 below:

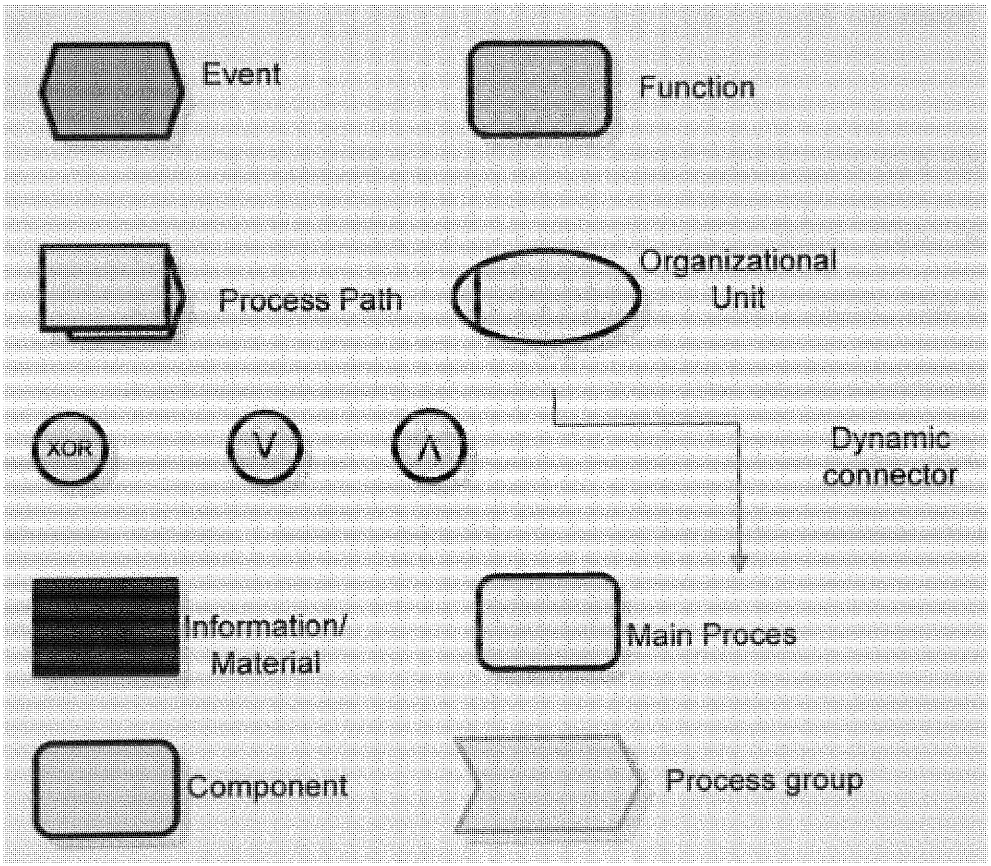


Figure 6: EPC objects

In analyzing the atomic tasks, the measure considers the following modularity facets in order to minimize dependence between modules and maximize cohesion between tasks:

a) Interface Independence defined by minimal dependencies that exist between component activities across modules, through their external interfaces. Each module uses independent inputs and makes independent contributions to the organization [42].

b) Tasks Dependence defined by strong couplings between component tasks that are grouped together in the same module. These require each other's inputs or outputs and share resources to be completed [42].

Based on the facets, the following comparison is explored and correlated to the modularity measure:

- Task – Task dependency, which occurs when tasks rely on each other with respect to their information, resource or control flows. These tasks are compatible with respect to the matching of outputs to inputs. That is, when a task needs an input or an output from previous task for completion.

Once task analysis is complete, a comparison of activity modules is done with a view to minimizing Activity – Activity dependencies to achieve loose couplings for process redesign.

3.4 MATHEMATICAL PROGRAMMING MODEL – CLUSTERING

The mathematical programming model is a set-partitioning integer nonlinear programming (INLP) problem [43-45] that seeks to determine the optimal grouping/assignment of all tasks in the workflow.

The decision variables in this model are binary integers that represent whether a task is assigned to module or not.

The INLP is constructed as a maximization problem that seeks to achieve the following two goals: 1) maximizes cohesion between tasks in a module and, 2) minimizes coupling between modules of the workflow. The model has two types of constraints:

1. A module time upper limit set by performance requirements for the workflow, gathered from time studies of the workflow
2. Interdependence relationships between tasks, determined from the DSM matrices of task interactions.

CHAPTER 4

4 MODULAR WORKFLOW DEVELOPMENT AND ANALYSIS

4.1 APPLICATION EXAMPLE

The example used throughout the next sections is the workflow of a telecommunications company, which is in the long-distance domestic and international market. The marketing strategy of the company is to focus on the growing US Hispanic market and international calls between the US and Latin America. This case study will focus on the prepaid calling card process. A calling card is a card in set denominations (eg. $10, $15, or $20) that customers purchase to make telephone calls. These cards can be used from any telephone and for any type of call.

The Event-driven process chain business modeling technique was used to model the prepaid calling card delivery process. The model of the process was created from documentation done by analysts who analyzed the As-Is process through observation, interviews and existing documentation.

Some keynotes on the process are the telecommunications company acts as a "coordinator" for the entire process. The cards are printed by an outside vendor, and distributed by an outside distributor who also collects payments, capacity on the network to carry the calls is negotiated with a provider, and customer service is outsourced. The company's actual task is the marketing of the prepaid cards. Figure 7 and Figure 8 shows the EPC diagram for the marketing of the card to customers and the entire prepaid card process. The process starts when the distributor submits a supply request and an inventory report. Sales then evaluates the request and if approved generates a personal identification number (PIN) order. The PIN is used to track each card. The Prepaid Manager then creates a CD-ROM of the PINs while marketing interacts with the Printer to manufacture cards. The cards are then printed and based on the subscribed customers the cards are activated.

Prepaid Calling Cards Process

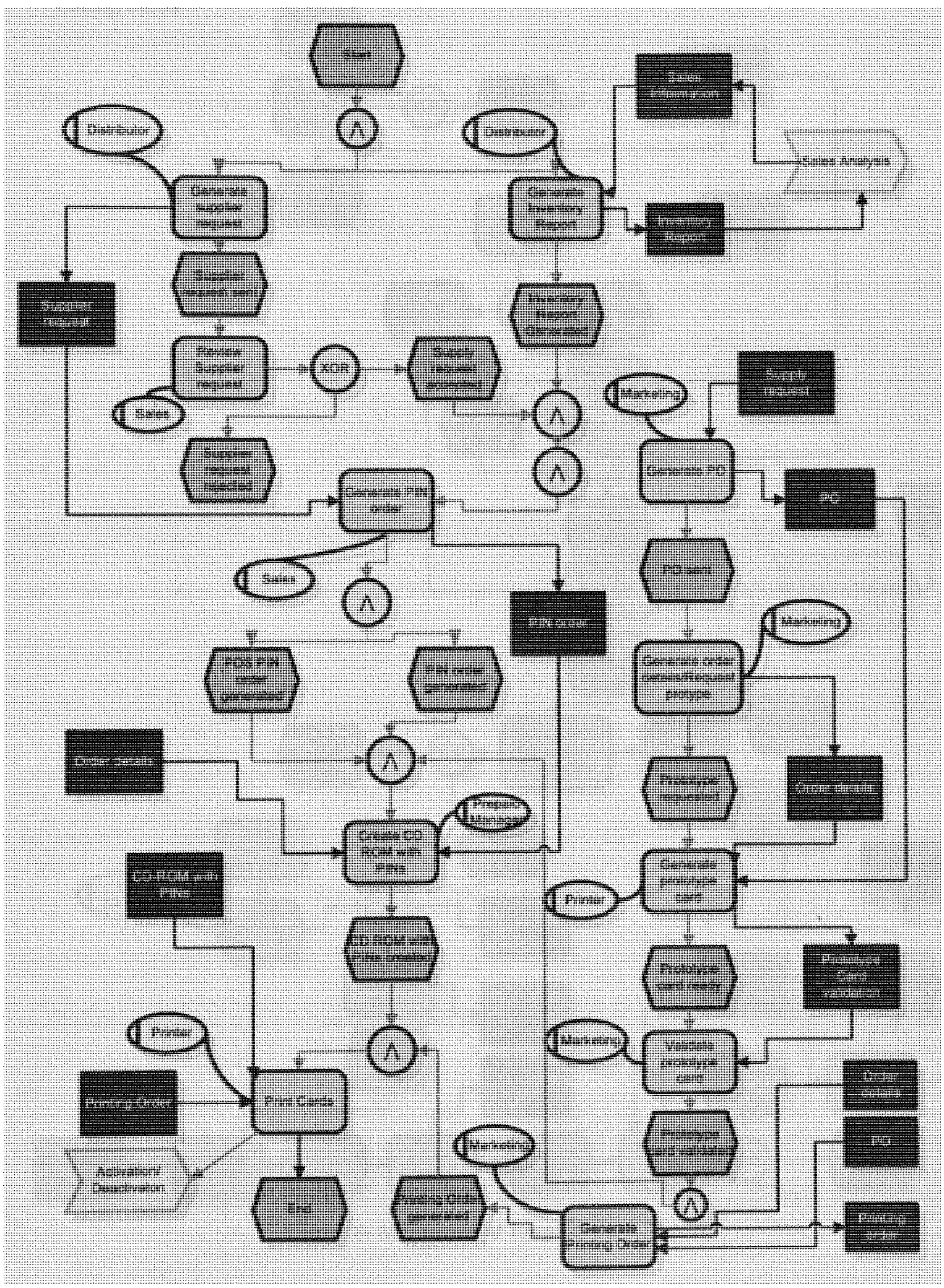


Figure 7: Prepaid Calling Card Process

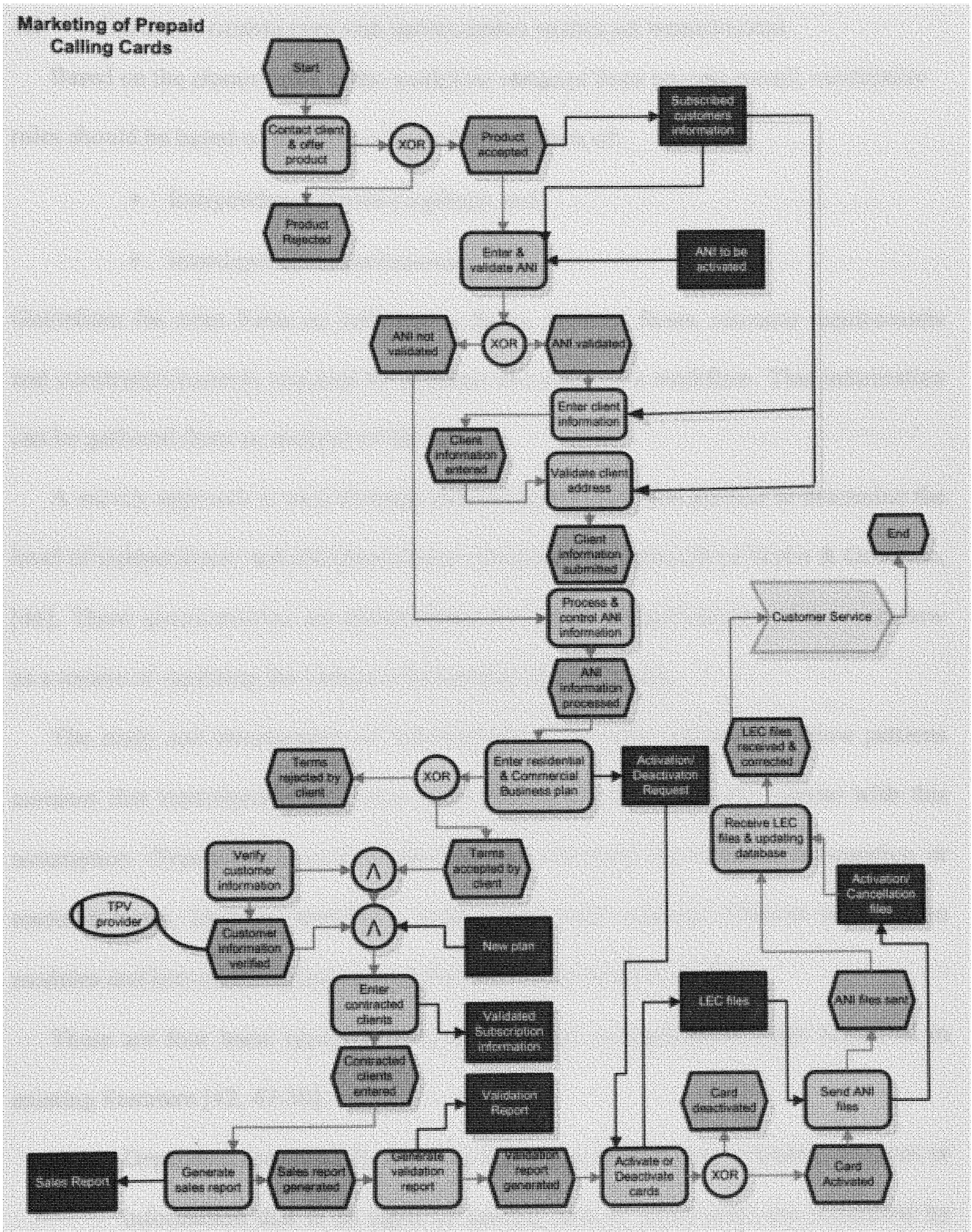

Marketing of Prepaid Calling Cards
Start
Contact client & offer product
XOR
Product accepted
Subscribed customers information
Product Rejected
Enter & validate ANI
ANI to be activated
ANI not validated
XOR
ANI validated
Enter client information
Client information entered
Validate client address
Client information submitted
End
Process & control ANI information
Customer Service
ANI information processed
LEC files received & corrected
Terms rejected by client
XOR
Enter residential & Commercial Business plan
Activation/ Deactivation Request
Verify customer information
Terms accepted by client
Receive LEC files & updating database
Activation/ Cancellation files
TPV provider
Customer information verified
New plan
Enter contracted clients
Validated Subscription information
LEC files
ANI files sent
Contracted clients entered
Validation Report
Card deactivated
Send ANI files
Sales Report
Generate sales report
Sales report generated
Generate validation report
Validation report generated
Activate or Deactivate cards
XOR
Card Activated

Figure 8: Marketing of Prepaid Card Process

4.2 General guidelines for developing modular workflows

Based on the atomic tasks of the workflow obtained from process model, modularity rules should be based on the general modularity criteria of:

- Independence (loose-coupling), and
- Interdependence (cohesion).

Guidelines for rules focus on information flows, material flows, resource requirements and constraints/controls required for efficient execution of a workflow. This information can be gathered from an analysis of the process.

A survey approach is used through questionnaires about the process to determine the level of independence and interdependence, similar to the approach of Wybo & Goodhue, [46]. These questionnaires are administered during the design/redesign of the workflow as a means of capturing the various relationships between tasks.

The study and measurement of interdependence through studying workflow patterns assumes that interdependencies arise from tasks [41]. This work will concur with this assumption. Dependencies exist when tasks are connected by resource, information or resource flows. Modular workflows must exhibit high cohesion between tasks within modules and low coupling and increased concurrency between modules.

There are four basic types of tasks relationships or dependency types identified in existing literature [42, 47-50].

a) Dependent (sequential), which is defined as when a task outputs a resource or information that is an input to another task or when tasks are connected by events.

b) Independent (concurrent or uncoupled), which refers to two tasks that occur in parallel and each task, is independent of the other.

c) Interdependent (coupled), which refers to a reciprocal dependency, when two tasks occur in parallel and each task, requires an input from the other task.

To explore the various types of relationships, the taxonomy in table 4 is used to differentiate the types of workflow interactions.

Table 4: Simple Taxonomy of Workflow Interactions

Information	Refers to the need for data, document or signal exchanges between two tasks, and defines the interface between tasks.
Resource	Refers to the need for people, databases, machines, etc. to enable exchanges between two tasks
Control	Refers to the conditions required to produce correct outputs, which include constraints, regulations, etc.

In order to properly measure and characterize the dependencies, attributes of the workflow will be defined. Giachetti [49] proposed an extended EPC model with attributes of frequency, importance and timing, which will be adopted here. These attributes are used to characterize the strength of dependence types for each workflow interaction. These dependence types are defined and represented in EPC as, pooled resources referring to functions that share resource; information sequential referring to a function whose output resource being the input to another function; control sequential referring to functions connected by events and/or connectors; and reciprocal to refer to two functions occurring in parallel where each requires an input from the other.

Information flow attributes are used to characterize the strength of pooled resources, information sequential and reciprocal dependencies, and control flow attributes are used to characterize the strength of control sequential dependencies.

Each interaction is quantified using a rating scheme that weighs interactions relative to each other and are evaluated based on responses to questions on the workflow process, as Table 5 illustrates:

Table 5: Workflow Interaction Quantification Scheme

Workflow Interaction	**Workflow Attribute**	**Scale**	**Interpretation**
Information/Resource	Frequency	1- Quarterly 2 - Monthly 3 - Biweekly 4 -Weekly 5 – Daily 6 – Hourly 7 – Minute	How frequently does this information flow/resource exchange occur?
	Importance	1 – Not important 3 – Dependent but not required to complete function	Is information/resource sharing between tasks important? How important is the flow of information/resource exchange between the

Workflow Interaction	Workflow Attribute	Scale	Interpretation
		5 – Very dependent and necessary for completion of function.	functions?
	Delay	1 – Quarterly or more 2 - Monthly 3 - Biweekly 4 -Weekly 5 – Daily 6 – Hourly 7 - Minute	If information/resource is unavailable, how long can the function be delayed?
Control	Importance	1 – Minor inconvenience 3 – Difficult but possible to function 5 – Impossible to function	How important are constraints/regulations to the execution of a function? How important is execution of a preceding function to the current function?

Workflow Interaction	**Workflow Attribute**	**Scale**	**Interpretation**
		properly	
	Delay	1 – Quarterly or more 2 - Monthly 3 - Biweekly 4 -Weekly 5 – Daily 6 – Hourly 7 - Minute	How long can the completion of preceding function be delayed negatively when controls are not followed?

Note the frequency is omitted for control flow because in control flow sequential interdependency the succeeding task is executed at the same frequency as the preceding task.

Other general questions to gather more information about the interactions between tasks in a workflow are listed below. These questions must be administered to people in the organization that are knowledgeable about the business process (Browning [51] for products). Questions include:

1) What inputs does each task need?
2) Where do these inputs come from (another tasks or out side the workflow)?
3) What outputs must each task produce?
4) Where do these outputs come from (another tasks or out side the workflow)?

5) Given an activity, what other activities must be executed before to provide information needed to execute this activity?
6) What activities can be executed without input from another activity?
7) If an activity cannot be performed, what other activities are affected, i.e. what activities cannot be executed?

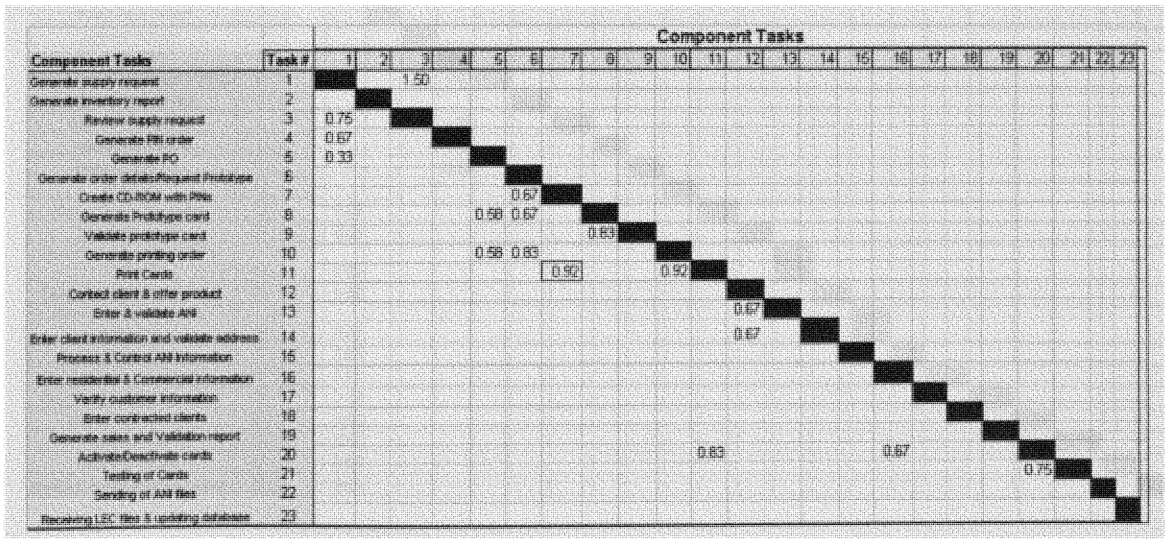
Independence relationships can be evaluated from the analysis of a component-based design structure matrix (DSM) approach [50, 52], where all n individual functions are compared in a $n \times n$ matrix and each element in the matrix represents weightings of each attribute representing the dependency type. Each dependency type will be analyzed separately.

This matrix formulation approach is applied broadly in manufacturing, and will be explored in the evaluation of task interdependencies in the next section.

4.3 Evaluation of Task Interdependencies

Decomposition of the workflow into blocks of tasks is made possible through use of matrices that enables the comparison of tasks based on their interdependencies. Based on the weightings of interdependence (from Table 5) between the workflow tasks (functions), *n x n* matrices are developed for each interdependence type. Each cell value represents the strength of the interdependence between the tasks. Each row depends on each column for each interdependence type and the values are sum of assigned importance and delay weightings, normalized to between 0 and 1. Note that in the example of the prepaid calling card process no reciprocal interdependence type was found, as this was an extract of the process. Tables 6, 7 and 8 show the interdependence matrices:

Table 6: Matrix of Information Sequential Interdependence Prepaid Calling Card Process



		Component Tasks																						
Component Tasks	Task #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Generate supply request	1	■		1.50																				
Generate inventory report	2		■																					
Review supply request	3	0.75		■																				
Generate PIN order	4	0.67			■																			
Generate PO	5	0.33				■																		
Generate order details/Request Prototype	6						■																	
Create CD-ROM with PINs	7						0.67	■																
Generate Prototype card	8					0.58	0.67		■															
Validate prototype card	9								0.83	■														
Generate printing order	10					0.58	0.83				■													
Print Cards	11							0.92			0.92	■												
Contact client & offer product	12												■											
Enter & validate ANI	13												0.67	■										
Enter client information and validate address	14												0.67		■									
Process & Control ANI information	15															■								
Enter residential & Commercial information	16																■							
Verify customer information	17																	■						
Enter contracted clients	18																		■					
Generate sales and Validation report	19																			■				
Activate/Deactivate cards	20											0.83					0.67				■			
Testing of Cards	21																				0.75	■		
Sending of ANI files	22																						■	
Receiving LEC files & updating database	23																							■

Table 7: Matrix of Control Sequential Interdependence for Prepaid Calling Card Process

		Component Tasks																						
Component Tasks	Task #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Generate supply request	1	■																						
Generate inventory report	2		■																					
Review supply request	3	0.75		■																				
Generate PIN order	4		0.42		■																			
Generate PO	5		0.42			■																		
Generate order details/Request Prototype	6					0.42	■																	
Create CD-ROM with PINs	7				0.67			■																
Generate Prototype card	8						0.67		■															
Validate prototype card	9								0.833	■														
Generate printing order	10									0.83	■													
Print Cards	11							0.92			0.92	■												
Contact client & offer product	12												■											
Enter & validate ANI	13												0.67	■										
Enter client information and validate address	14														■									
Process & Control ANI information	15													0.67		■								
Enter residential & Commercial information	16																■							
Verify customer information	17																0.50	■						
Enter contracted clients	18																	0.50	■					
Generate sales and Validation report	19																		0.75	■				
Activate/Deactivate cards	20																				■			
Testing of Cards	21																				0.75	■		
Sending of ANI files	22																						■	
Receiving LEC files & updating database	23																							■

Table 8: Matrix of Pooled Resources Interdependence for Prepaid Calling Card Process

		Component Tasks																						
Component Tasks	Task #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Generate supply request	1	■																						
Generate inventory report	2	0.83	■																					
Review supply request	3			■																				
Generate PIN order	4				■																			
Generate PO	5					■																		
Generate order details/Request Prototype	6					0.83	■																	
Create CD-ROM with PINs	7							■																
Generate Prototype card	8					0.58			■															
Validate prototype card	9					0.58				■														
Generate printing order	10					0.58					■													
Print Cards	11											■												
Contact client & offer product	12		0.83										■											
Enter & validate ANI	13		0.83											■										
Enter client information and validate address	14		0.83												■									
Process & Control ANI information	15		0.83													■								
Enter residential & Commercial information	16		0.83														■							
Verify customer information	17		0.83															■						
Enter contracted clients	18		0.83																■					
Generate sales and Validation report	19		0.83																	■				
Activate/Deactivate cards	20		0.83																		■			
Testing of Cards	21		0.83																			■		
Sending of ANI files	22		0.83																				■	
Receiving LEC files & updating database	23		0.83																					■

By examining the strength of the dependencies between tasks, the tasks can be clustered into sub-groupings.

4.4 CLUSTER ANALYSIS

In general the main objectives of workflow design and redesign is to decrease throughput time required to handle cases, decrease the required cost of executing the workflow, improving the quality of service delivered, and improve the ability of the workflow to react to variations [36].

In clustering to form process modules the foremost consideration is to maximize interactions between tasks within clusters (modules), while minimizing interactions between modules. Clustering is done in two phases [52]:

1) Composite analysis of the various types of interdependencies and identifying the sequence of tasks and the structure of the workflow.
2) Clustering of coupled tasks and establishing criteria for the management of all interdependencies.

Phase1: Composite Analysis of Interdependencies

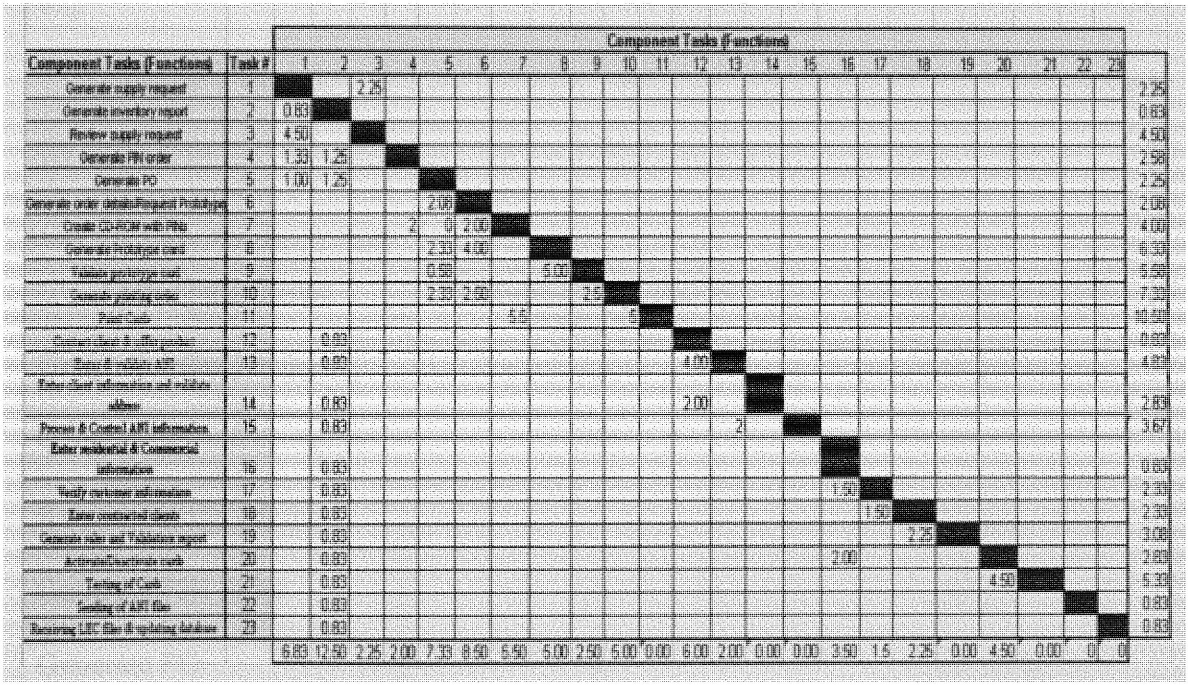
It is expected that most workflows be ordered sequentially because of the ease of managing these flows as well as the quality associated with these flows. As such information sequential interdependencies are considered easier to manage than pooled resources, control sequential and reciprocal interdependencies. Reciprocal interdependencies are the most difficult to manage in any organization as they require more coordination effort [49], and are therefore considered the most important in workflow design. These relationships are generally kept in the same module. Therefore in my analysis the following weightings are given to each interdependence type based on their relative importance, as shown in Table 9 below.

Table 9: Relative Importance of Interdependency type

Interdependency Type	**Relative Weighting**
Pooled Resource	1
Information Sequential	3
Control Sequential	3
Reciprocal	5

Based on the relative importance weightings, the matrices of task interdependencies were added to obtain a single 2-Dimensional matrix as shown in Table 10.

Table 10: Composite Matrix of Tasks Interdependencies in the Workflow (Un-partitioned)



		Component Tasks (Functions)																							
Component Tasks (Functions)	Task #	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
Generate supply request	1	■		2.25																					2.25
Generate inventory report	2	0.83	■																						0.83
Review supply request	3	4.50		■																					4.50
Generate PIN order	4	1.33	1.25		■																				2.58
Generate PO	5	1.00	1.25			■																			2.25
Generate order details/Request Prototype	6					2.08	■																		2.08
Create CD-ROM with PINs	7				2	0	2.00	■																	4.00
Generate Prototype card	8					2.33	4.00		■																6.33
Validate prototype card	9					0.58			5.00	■															5.58
Generate printing order	10					2.33	2.50			2.5	■														7.33
Print Cards	11							5.5			5	■													10.50
Contact client & offer product	12		0.83										■												0.83
Enter & validate ANI	13		0.83										4.00	■											4.83
Enter client information and validate address	14		0.83										2.00		■										2.83
Process & Control ANI information	15		0.83											2		■									3.67
Enter residential & Commercial information	16		0.83														■								0.83
Verify customer information	17		0.83														1.50	■							2.33
Enter contracted clients	18		0.83															1.50	■						2.33
Generate sales and Validation report	19		0.83																2.25	■					3.08
Activate/Deactivate cards	20		0.83														2.00				■				2.83
Testing of Cards	21		0.83																		4.50	■			5.33
Sending of ANI files	22		0.83																				■		0.83
Receiving LEC files & updating database	23		0.83																					■	0.83
		6.83	12.50	2.25	2.00	7.33	8.50	5.50	5.00	2.50	5.00	0.00	6.00	2.00	0.00	0.00	3.50	1.5	2.25	0.00	4.50	0.00	0	0	

The structure of the matrix is also examined, in terms of tightly coupled tasks (reciprocal relationships) and any other tasks that can be operated concurrently. However, for the example workflow no reciprocal relationships were identified. In the actual workflow there are several reciprocal relations that must be identified in this phase of the clustering. The procedure is adopted from [47].

Phase 2: Modularization Model - INLP

The composite matrix of task interdependencies is used to cluster tasks into modules. The clustering process seeks to group strongly coupled tasks together due to their complexity and separating weakly coupled tasks. The model focuses on the strength of the interdependencies between tasks and the organizational units responsible for each task. This is keeping with the view that interdependencies between tasks create a coordination load on the organizational unit or actor responsible coordinating the tasks [49].

We define a workflow that can perform i tasks that are assigned into j modules. Each task takes time t_i to be completed within a module, as illustrated in Table 11 below:

Table 11: Tasks Throughput times

Task (Functions)	**Throughput Time (t_i) (mins)**	**Initial Department Assignment**
Generate supply request	5	Distributor
Generate inventory report	10	Distributor
Review supply request	5	Sales
Generate PIN order	2	Sales
Generate PO	2	Marketing
Generate order details/Request Prototype	10	Marketing
Create CD-ROM with PINs	5	Prepaid Manager

Task (Functions)	**Throughput Time (t_i) (mins)**	**Initial Department Assignment**
Generate Prototype card	15	Marketing
Validate prototype card	5	Marketing
Generate Printing order	5	Marketing
Print Cards	30	Printer
Contact client & offer product	5	Sales
Enter & validate ANI	5	Sales
Enter client information and validate address	5	Sales
Process & Control ANI information	10	Sales
Enter residential & Commercial information	5	Sales
Verify customer information	5	Sales
Enter contracted clients	2	Sales
Generate sales and Validation report	5	Sales
Activate/Deactivate cards	5	Sales
Testing of Cards	5	Sales
Sending of ANI files	30	Sales
Receiving LEC files & updating database	10	Sales
Total Workflow Time (W)	186	

Only one task can be completed at a time. At the start of the process there are m tasks to be completed. The time taken to complete m tasks in a module T_j must be less than the total workflow time (W) (i.e. $T_j \leq W$). Each task i require multiple data elements (interfaces) defined by the strength of the interactions between tasks. Let I_{ik} represents the interdependence strength between tasks i and tasks k. To formulate the ILP model to determine the optimal tasks groupings and sequencing, the following constraints will be considered:

1. Tasks are assigned to be completed one at a time to modules.
2. A module can be assigned zero or many tasks
3. Tasks with the strongest interdependences are considered first for assignment to a module. If a module has no room for the task being considered, i.e. $T_j \leq W$, a new module must be started (called the First-Fit Decreasing Heuristic).

We define the following decision variables to capture the grouping/assignment of tasks:

$$X_{ij} = \begin{cases} 1, \text{ if task } i \text{ is assigned to module } j \\ 0, \text{ otherwise} \end{cases}$$

$X_{11} = 1$, which represents the fact that we are starting with task 1 in module 1.

$i = 1, 2, \ldots\ldots\ldots m$ tasks

$j = 1, 2, \ldots\ldots\ldots N$ modules, where $N \geq$ number of organizational units in the workflow.

Since all tasks need to be processed,

$$\sum_{j=1}^{N} X_{ij} = 1 \qquad \forall i$$

$$\sum_{j=1}^{N} X_{ij} \leq T_{j\,Averages} \qquad \forall j$$

$T_{javerage} = W/|j|$

$T_{javerage} \leq T_{j\,Desiredflowtime}$

Therefore the objective function is:

1) To maximize cohesion between tasks within a module,

$$\text{Maximize Z} = \sum_{j=1}^{N}\sum_{i=1}^{m}\sum_{k=1}^{m} X_{ij} * I_{ik} * X_{kj}$$

2) To minimize couplings between modules

$$\text{Minimize Z} = \sum_{j=1}^{N}\sum_{i=1}^{m}\sum_{k=1}^{m}\sum_{l \neq j} X_{ij} * X_{kl} * I_{ik}$$

Where $k = 1, 2, \ldots\ldots m$, and represents tasks compared to other task i, $(i > k)$, within and between modules.

To obtain one objective function,

$$\text{Maximize Z} = \sum_{j=1}^{N}\sum_{i=1}^{m}\sum_{k=1}^{m} X_{ij} * I_{ik} * X_{kj} - \sum_{j=1}^{N}\sum_{i=1}^{m}\sum_{k=1}^{m}\sum_{l \neq j}^{m} X_{ij} * X_{kl} * I_{ik}$$

The overall formulation becomes,

$$\text{Maximize Z} = \sum_{i=1}^{m}\sum_{j=1}^{N}\sum_{k=1}^{m} X_{ij} * I_{ik} * X_{kj} - \sum_{i=1}^{m}\sum_{j=1}^{N}\sum_{k=1}^{m}\sum_{l \neq j}^{m} X_{ij} * X_{kl} * I_{ik}$$

Subject to $\sum_{j=1}^{N} X_{ij} = 1 \qquad \forall i$

$\sum_{j=1}^{N} X_{ij} \leq T_{j\,Averages} \qquad \forall j$

$T_{javerage} = W/|j|$

$T_{javerage} \leq T_{j\,Desiredflowtime}$

$$X_{ij} \in \{0,1\},\ i > k\ ,\ l \neq j$$

$$l = j+1,\ldots$$

The model is then tested in LINGO to obtain the optimal task assignment to modules.

The INLP model is illustrated in LINGO as:

```
MODEL:
SETS:

TASK /T1..T7/: TIME;
MODULE /M1..M7/:;
TxT (TASK, TASK): D;
TXM (TASK, MODULE): X;

ENDSETS

MAX = COH - COUP;

COH = @SUM(MODULE(J):
            @SUM(TASK(I):
                  @SUM(TASK(K)|K#NE#I: X(I,J)*X(K,J)*D(I,K))));

COUP= @SUM(MODULE(J)|J#LE#2:
            @SUM(MODULE(L)|L#GT#J:
                  @SUM(TASK(I):
                        @SUM(TASK(K)|K#GT#I: X(I,J)*X(K,L)*D(I,K)))));
! Constraints;

@FOR(TASK(I):
            @SUM(MODULE(J): X(I,J))=1);

@FOR(MODULE(J):
      @SUM(TASK(I): X(I,J)*TIME(I)) <= 35);

@FOR(T x M: @BIN(X));

DATA:
D =
0     0     0     0     1     1     0
0     0     0     1     0     0     1
0     1     0     1     0     0     1
0     1     1     0     1     0     1
0     0     0     1     0     1     0
1     0     0     0     1     0     0
0     1     1     1     0     0     0;

TIME = 20 10 5 5 5 10 5;
ENDDATA
END
```

The following steps summarize the approach:

Step 0. Workflow Decomposition: Breakdown the workflow into atomic tasks, then compare each task to every other tasks based on the workflow interaction type and quantification for all workflow attributes.

Step 1: Task Interdependency Analysis: Develop task-task interaction matrices for the three interdependence types (Matrix *A, B & C*), showing interdependence strengths between tasks as cell values. The interdependence strengths are the sum of the weighs of the all attributes of an interaction.

Step 2. Composite Interdependence Analysis: Add interaction matrix for each interdependence type to obtain a composite matrix of task interdependencies (Matrix *D*). Also identify any reciprocal relationships in matrix *D* for optimizing of task assignment. These tasks must be in the same module.

Step 3. Modularization: Based on data of task times and average desirable workflow time obtained from time studies of the workflow, and data on interdependence strengths, apply INLP modularization model. Input model and data into Lingo and run model.

Step 4: Classification: Collect model output results and analyze optimal task assignments to identify modules.

Step 5: Stop and create modularity matrix from output results.

Step 6: Review modularity matrix to ensure highly coupled tasks (reciprocal relationships) are in the same module.

4.5 TESTING AND VALIDATION OF THE MODULARIZATION MODEL

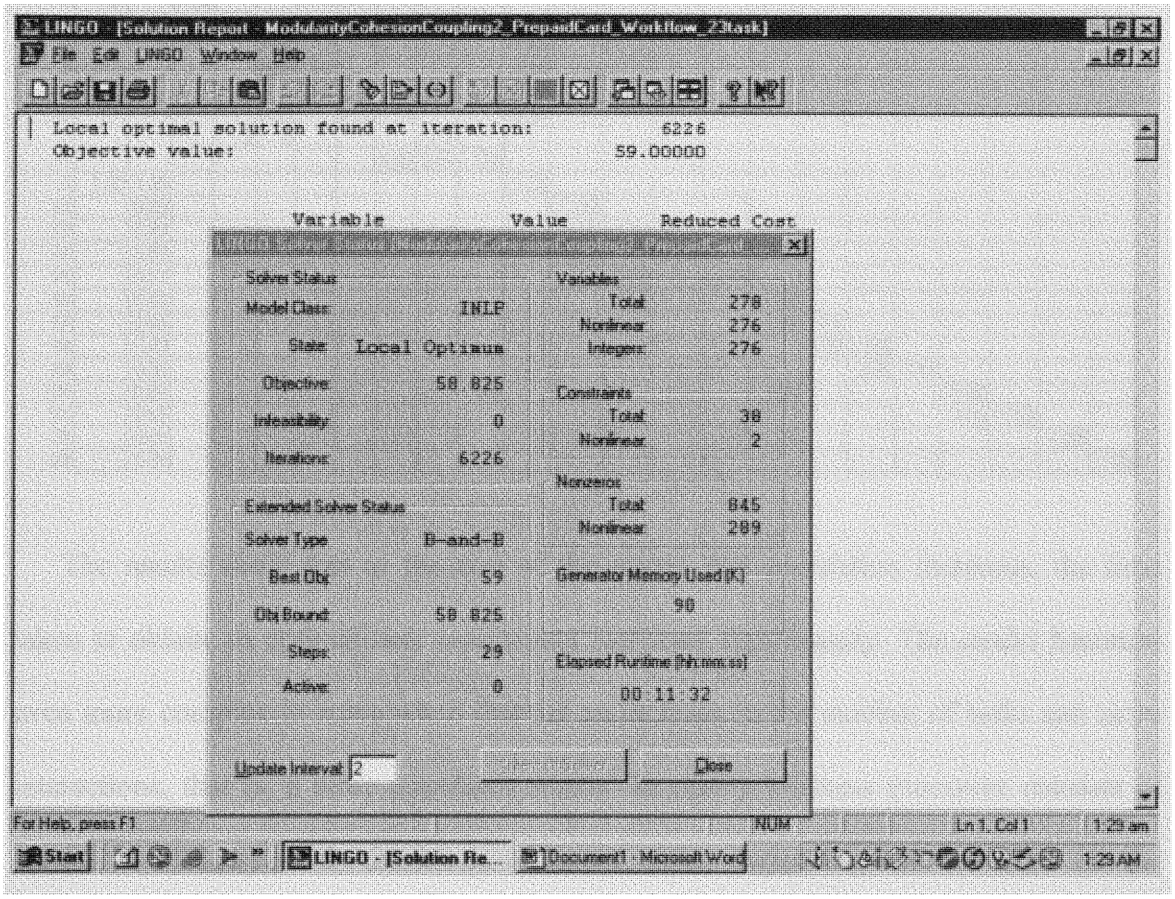
To test the model various size DSM for several product designs is inputted in LINGO model and the results compared to known solutions of modularity matrices done by Browning, Kusiak and Yassine [51, 53, 54]. Each time INLP model is run in LINGO the solving time and the optimal task assignment were observed and compared. The population of the matrix was also considered as a factor in comparing solution time. This is calculated based on the size of the matrix and cell entries for task interactions. Table 12 illustrates the results obtained. Appendix 1 shows the LINGO output results and task assignments obtained, which in most cases exactly matched the modules obtained by other methods [51, 53, 54]. In other cases, such as the 14, 16 and 22 tasks comparisons the tasks assignments to modules varied as the number of tasks increased and the matrices were more populated.

Table 12: Model Validation – Comparison of known solutions

Number of tasks	**Number of Modules**	**Matrix population (<15% considered sparse)**	**Time to solve**
5	2	3%	2 seconds
7	2	35%	1 seconds
11	7	16%	13 seconds
14	6	20%	34 seconds
16	9	9%	14 seconds
22	9	37%	1 hr 48 minutes
27	18	49%	9 hrs 13 minutes*

* Note Lingo solver was interrupted at this time, no optimal solution reached.

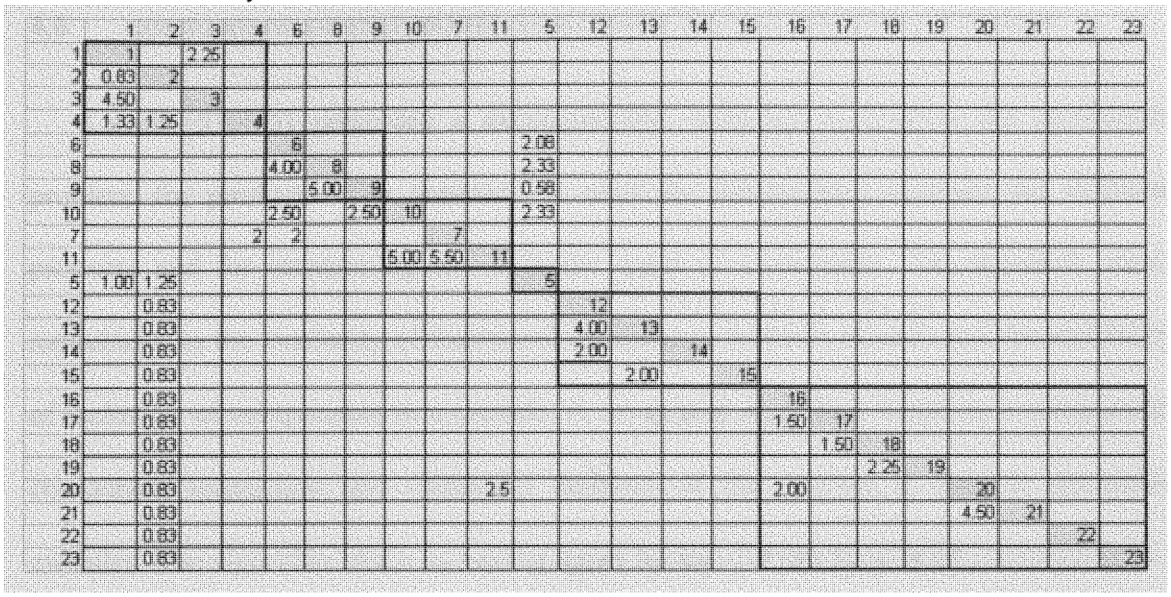
The results of INLP model for the prepaid calling card example, which 8% populated, is shown Table 13 below:


LINGO - [Solution Report - ModularityCohesionCoupling2_PrepaidCard_Workflow_23task]
File Edit LINGO Window Help
For Help, press F1
NUM
Ln 1, Col 1
1:29 am
Start
LINGO - [Solution Re...
Document1 - Microsoft Word
1:29 AM

```
Local optimal solution found at iteration:          6226
Objective value:                                59.00000


                Variable           Value        Reduced Cost
```

```
Solver Status                          Variables
Model Class:              INLP                  Total:        278
                                            Nonlinear:        276
       State:   Local Optimum                Integers:        276

   Objective:           58.825         Constraints
                                                Total:         38
  Infeasibility:             0              Nonlinear:          2

  Iterations:             6226         Nonzeros
                                                Total:        845
Extended Solver Status                      Nonlinear:        289
Solver Type:            B-and-B
                                       Generator Memory Used (K)
    Best Obj:               59                     90

   Obj Bound:           58.825
                                       Elapsed Runtime (hh:mm:ss)
       Steps:               29
                                            00:11:32
      Active:                0

Update Interval: 2                                         Close
```

Table 13: Modularity Matrix for Prepaid Calling Card example – INLP Model results



	1	2	3	4	6	8	9	10	7	11	5	12	13	14	15	16	17	18	19	20	21	22	23
1	1		2.25																				
2	0.83	2																					
3	4.50		3																				
4	1.33	1.25		4																			
6					6						2.08												
8					4.00	8					2.33												
9						5.00	9				0.58												
10					2.50		2.50	10			2.33												
7				2	2				7														
11								5.00	5.50	11													
5	1.00	1.25									5												
12		0.83										12											
13		0.83										4.00	13										
14		0.83										2.00		14									
15		0.83											2.00		15								
16		0.83														16							
17		0.83														1.50	17						
18		0.83															1.50	18					
19		0.83																2.25	19				
20		0.83								2.5						2.00				20			
21		0.83																		4.50	21		
22		0.83																				22	
23		0.83																					23

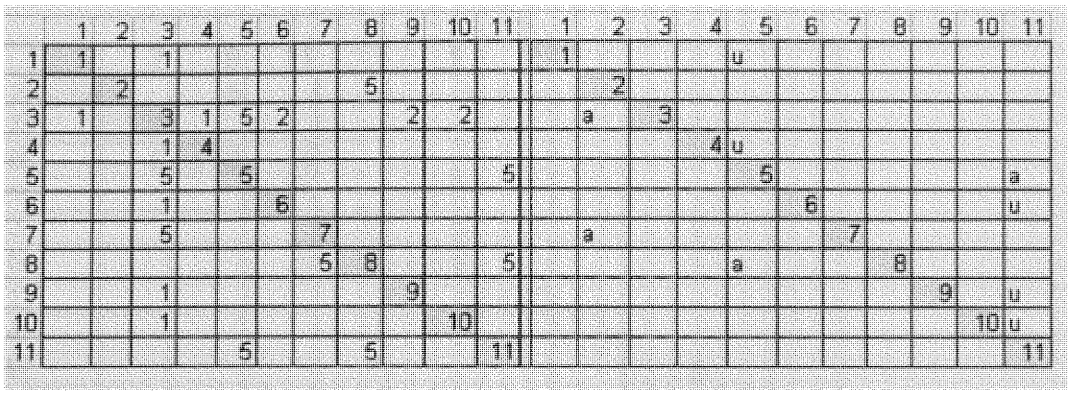
The results show that as the size of the matrix and population increases, the time taken to obtain an optimal task assignment increases significantly. The model becomes even more complex to solve when there is an increase in the number of tasks and more interactions, that is, a more populated matrix. In reality business processes consists of several workflows with numerous different tasks depending on the size of the organization, therefore designing for modularity with this model would be an inefficient process, especially considering the fact that several iterations would be required to obtain the most efficient assignment and sequencing of tasks.

It is with this in mind that we compare the model to Kusiak's [53] decomposition algorithm for product design and recommend a heuristic approach in the next sections 4.6 and 4.7, respectively.

4.6 COMPARISON OF INLP MODEL TO KUSIAKS' DECOMPOSITION APPROACH

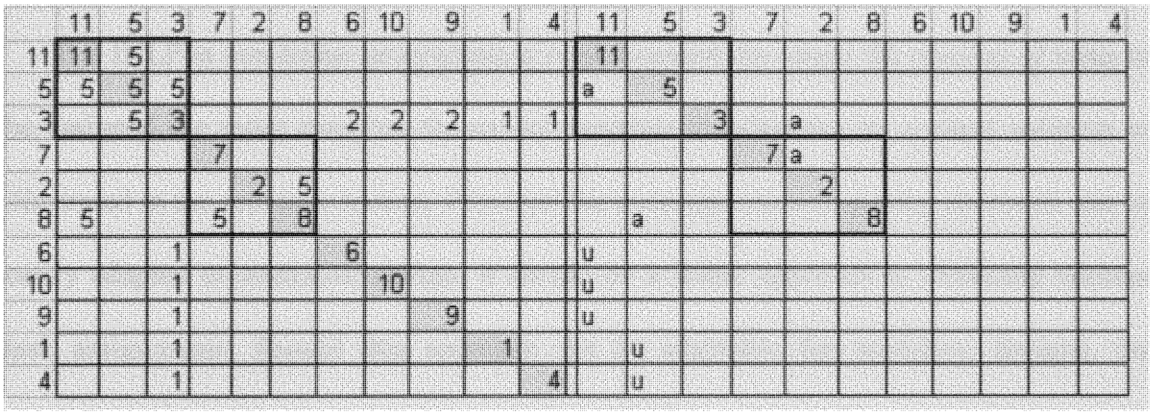
Kusiak [53, 55] presented a decomposition approach to address the modularity problem for products. The approach is used for determining modules for different products and for interpreting the different types of modularity. The approach views modularity as the similarity of functional interactions within a module and the suitability of inclusion of components in a module. A seven-step algorithm based on a component-component interaction and suitability matrices of the product is used to determine modules. The algorithm starts by setting upper bounds for the number of components in a module and the cost of duplicating components. It then triangularize the interaction matrix [55], rearranges the suitability matrix, then combines the two matrices to identify modules. It continues optimizing by deleting and duplicating components based on set conditions, and then analyzes the modularity matrix to classify modules according to three axioms, for component-swapping modularity, component sharing modularity and bus modularity, respectively. The approach is used at both the conceptual design and detail design phases for optimal formation of modules and is applicable even in situations where insufficient information is available. Table 14(a) and (b) below shows the results for a desk lamp.

Table 14 (a): Interaction and Suitability Matrix for Desk Lamp



	1	2	3	4	5	6	7	8	9	10	11	1	2	3	4	5	6	7	8	9	10	11
1	1		1									1				u						
2		2						5					2									
3	1		3	1	5	2			2	2			a	3								
4			1	4											4	u						
5			5		5						5					5						a
6			1			6											6					u
7			5				7						a					7				
8							5	8			5					a			8			
9			1						9											9		u
10			1							10											10	u
11					5			5			11											11

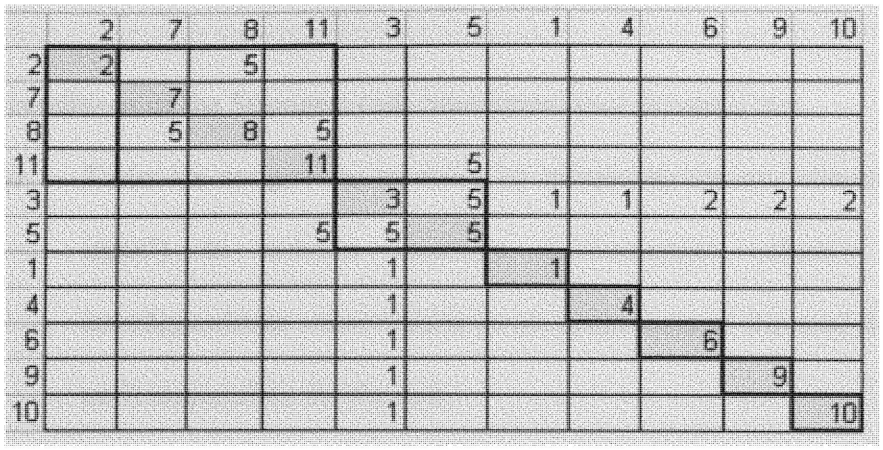
Table 14 (b): Modularity Matrix for Desk Lamp



	11	5	3	7	2	8	6	10	9	1	4	11	5	3	7	2	8	6	10	9	1	4
11	11	5										11										
5	5	5	5									a	5									
3		5	3				2	2	2	1	1			3		a						
7				7											7	a						
2					2	5										2						
8	5			5		8							a				8					
6			1				6					u										
10			1					10				u										
9			1						9			u										
1			1							1			u									
4			1								4		u									

Compared to the clustering approach presented in this thesis, the above approach seems less complex, and therefore easier to obtain optimal modules. It is a straightforward approach that allows design of various products with several components at different stages in the design process. The INLP model although it obtained a comparable optimal task assignment in 13 seconds for the same desk lamp example, as shown in Table 15, it will fail to efficiently create modules in real applications when the number of tasks increases.

Table 15: Modularity Matrix from INLP Model for Desk Lamp



	2	7	8	11	3	5	1	4	6	9	10
2	2		5								
7		7									
8		5	8	5							
11				11		5					
3					3	5	1	1	2	2	2
5				5	5	5					
1					1		1				
4					1			4			
6					1				6		
9					1					9	
10					1						10

The INLP is non-linear programming hard, because the larger the number of tasks to be assigned the more complex the model becomes in achieving an optimal solution as illustrated in the previous section. Provided the decomposition clustering approach can be easily adapted to workflow design, it should be a better approach to obtaining optimal task assignments. Should this approach not be adaptable for workflow designs, the heuristic presented in the next section can be applied.

4.7 HEURISTIC DEVELOPMENT

The heuristic is based on the assignment of task to modules based on the interdependence strengths, and is referred to as the Strongest-task-interdependence heuristic. Tasks are balanced to modules N ($N \geq$ organizational units) considering the interdependence strength between the tasks. Here tasks are added to organizational units to be completed based on interdependence strength and the responsible organizational unit. If the interdependence strength (I_{ik}) between tasks i and task k is greater than 1.67 ($I_{ik} \geq 1.67$), assign task to module. If a decision must be made between two or more tasks then the one with the strongest interdependence is added to relevant organizational unit.

The conditions of the Strongest-task-interdependence heuristic are:

a) Tasks are assigned to modules by comparing the row elements ('information input') to column elements ('information output'). The strongest interdependence value results in an assignment to the module (organizational unit) responsible for the task.

b) Assignment begins with task $i = 1$ in module $j = 1$

c) Highly coupled and reciprocal tasks are assigned to the same module. If both entries ij and ji are filled this indicates two-interdependency or coupling between the tasks. These tasks are considered to be complex and are assigned to the same module. They are assigned first.

d) Interdependence strength must be greater than zero to be considered for assignment. Note, if tasks in rows (and corresponding columns) i and j of the interdependence matrix have no direct interfaces, then this indicates independence (concurrency). Therefore these tasks are grouped into separate modules.

e) The average time to complete all assigned tasks to a module (T_{javg}) should not exceed workflow cycle time (W). Note T_{javg} and W will be inputted based on the specific workflow being modeled. Once the average workflow time is exceeded, a new module is started.

The procedure for the heuristic is as follows:

Step 1: Develop an interdependence matrix of all tasks to be performed to complete the workflow.

Step 2: Identify organizational units responsible for each task.

Step 3: Determine average workflow cycle time from time studies of the workflow.

Step 4: Assign tasks to modules based on the above conditions.

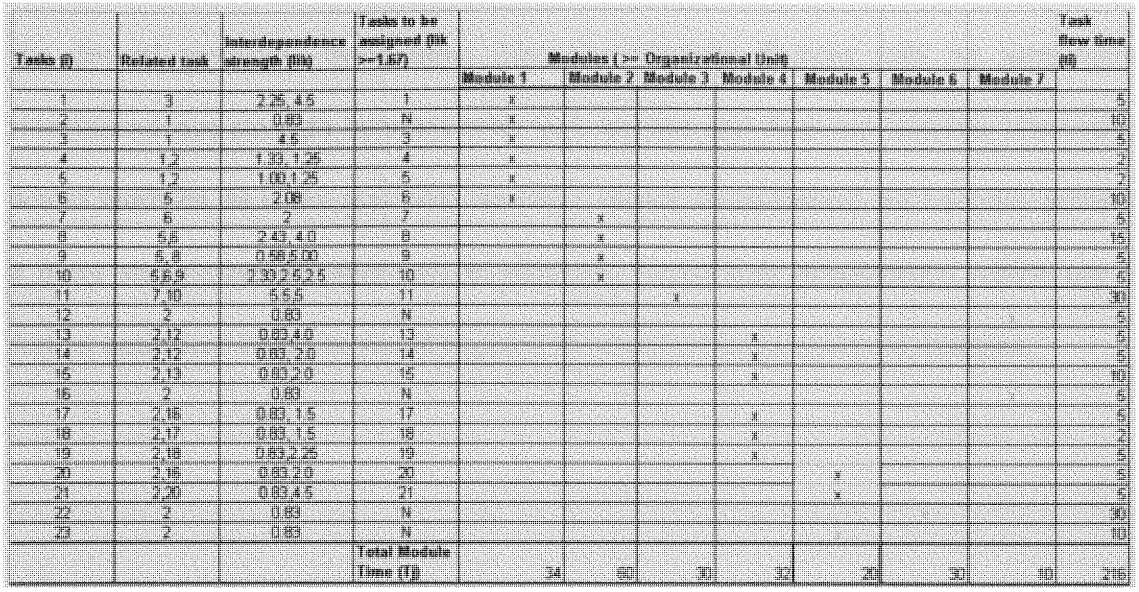
a) Starting with task i = 1 and module j = 1, determine tasks to be assigned based on the interdependence strengths. Highly coupled and reciprocal tasks are assigned first.

b) Assign the task from the list with highest interdependence strength and calculate total times of tasks already assigned to the module to ensure workflow cycle time is not exceeded.

Step 5: Close assignment of tasks to module i once module time exceeds workflow cycle time. Set $i = i + 1$ and go back to step 4. If there are no more unassigned tasks, the procedure is complete.

Therefore based on the heuristic, and assuming a desired module time of 35 minutes for the 186 minutes workflow outlined in table 11 above, the initial task assignment shown in Table 16 is obtained:

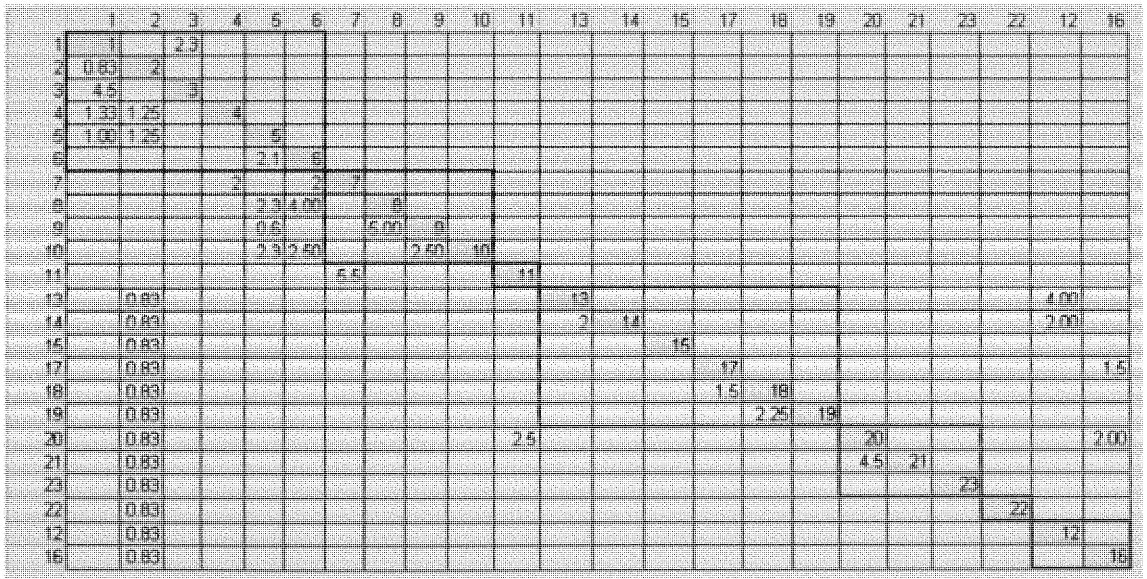
Table 16: Initial Task Assignment - Heuristic



Tasks (i)	Related task	Interdependence strength (Iik)	Tasks to be assigned (Iik >=1.67)	Modules (>= Organizational Unit)							Task flow time (ti)
				Module 1	Module 2	Module 3	Module 4	Module 5	Module 6	Module 7	
1	3	2.25, 4.5	1	x							5
2	1	0.83	N	x							10
3	1	4.5	3	x							5
4	1,2	1.33, 1.25	4	x							2
5	1,2	1.00,1.25	5	x							2
6	5	2.08	6	x							10
7	6	2	7		x						5
8	5,6	2.43, 4.0	8		x						15
9	5, 8	0.58,5.00	9		x						5
10	5,6,9	2.33,2.5,2.5	10		x						5
11	7,10	5.5,5	11			x					30
12	2	0.83	N								5
13	2,12	0.83,4.0	13				x				5
14	2,12	0.83, 2.0	14				x				5
15	2,13	0.83,2.0	15				x				10
16	2	0.83	N								5
17	2,16	0.83, 1.5	17				x				5
18	2,17	0.83, 1.5	18				x				2
19	2,18	0.83,2.25	19				x				5
20	2,16	0.83,2.0	20					x			5
21	2,20	0.83,4.5	21					x			5
22	2	0.83	N								30
23	2	0.83	N								10
			Total Module Time (Tj)	34	60	30	32	20	30	10	216

The modularity matrix is as shown in table 17 below.

Table 17: Modularity Matrix for prepaid calling card: Heuristic approach



	1	2	3	4	5	6	7	8	9	10	11	13	14	15	17	18	19	20	21	23	22	12	16
1	1		2.3																				
2	0.83	2																					
3	4.5		3																				
4	1.33	1.25		4																			
5	1.00	1.25			5																		
6					2.1	6																	
7				2		2	7																
8					2.3	4.00		8															
9					0.6			5.00	9														
10					2.3	2.50			2.50	10													
11							5.5				11												
13		0.83										13										4.00	
14		0.83										2	14									2.00	
15		0.83												15									
17		0.83													17								1.5
18		0.83													1.5	18							
19		0.83														2.25	19						
20		0.83									2.5							20					2.00
21		0.83																4.5	21				
23		0.83																		23			
22		0.83																			22		
12		0.83																				12	
16		0.83																					16

The results of the heuristic approach gave 7 modules, (1,2,3,4,5,6), (7,8,9,10), (11), (13,14,15,17,18,19), (20,21,23), (22) and (12,16). The INLP model obtained 6 modules, (1,2,3,4), (6,8,9), (10,7,11), (5), (12,13,14,15) and (16,17,18,19,20,21,22,23). The results are comparable, however difference could be due to the subjectivity of tasks times and the desired workflow cycle time. Also the heuristic focuses only on maximizing interdependence strength between tasks in a module and does not consider interactions across modules, as is the case with the INLP model. Nevertheless, the heuristic provides a more straightforward approach to obtaining initial task assignments, which can be further optimized based on the strength of observed interactions across modules.

CHAPTER 5

5 MEASURING MODULARITY IN THE WORKFLOW

The modularity measure used for workflows adopted concepts from the efforts of Gershenson et al. [6], as outlined in Section 2.1 of the literature review. The measure was chosen because it focused on creating modules that encourage independence between components and all life-cycle processes in different modules, similarity in components and processes in a module. The measure considers the module; it's components and tasks, and the interactions between them to measure similarity and independence. Independence referring to the minimal interactions between components in modules and similarity to capture those components in the module that are processed in a similar manner, as outlined below:

$$\text{Modularity} = [S_{in} /(S_{in} + S_{out})] + [D_{in}/(D_{in} + D_{out})]$$

Where, S_{in} = similarity between components in a module

S_{out} = similarity between different modules

D_{in} = dependency between components in a module

D_{out} = dependency between different modules

In keeping with task-oriented view of workflows, an adaptation of this measure will enable the analysis of workflows through distinguishing between the structure of a workflow and dynamic state, the tasks and interdependencies between tasks. The following will outline the adopted measure for workflows.

For a high degree of modularity, it is important to have a high degree of interdependence (coupling) between component tasks within a module (D_{in}) and minimal dependencies between modules (D_{out}). Therefore since the focus is now on loose-couplings between modules the relative modularity presented focuses on tasks interdependencies within and across modules.

The relative modularity measure applied is as shown below:

Relative Workflow Modularity = $D_{in} / (D_{in} + D_{out})$

D_{in} uses the ratings of task-task comparisons within a module for all interaction types. These values will be taken straight from the modularized matrix. D_{in} will have a positive effect on the measure, as it's important to group dependent tasks.

D_{in} is defined here to refer to the dependencies between each task within a module defined by the sum of pooled resource interactions, information sequential interactions and control sequential interactions.

$$D_{in} = \sum_{m=1}^{M} \sum_{i=r}^{s-1} \sum_{k=i+1}^{s} (D_{ik} + D_{ki})$$

Where:

m = is a module

M = number of modules in the workflow

r = first component task in the module m

s = last component task in the module m

i, k are component task in the same module

D_{ik} is the dependence between component task i and task k

D_{out} uses the ratings of task-task interactions for each component task compared to another component task outside a module for all interaction types. D_{out} has a negative impact on the total measure, as all external dependencies must be minimized to have independent modules.

$D_{out,}$ is defined here to refer to the dependencies between component task of a module and each component tasks that are external to the module,

$$D_{out} = \sum_{m=1}^{M-1}\sum_{i=r_1}^{s-1}\sum_{n=m+1}^{M}\sum_{k=r_2}^{s}(D_{ik} + D_{ki})$$

Where,

i, k are component task not in the same module

D_{ik} is the dependence between component task *i* and task *k*

The measure is applied to the modularized workflow obtained using the INLP model in table 13 and that obtained using the heuristic in table 17. Outlined below is the calculation for the relative modularity for each case.

1. **Relative Modularity for the Modularized Workflow – INLP Model**

$$D_{in} = \sum_{m=1}^{M}\sum_{i=r}^{s-1}\sum_{k=i+1}^{s}(D_{ik} + D_{ki})$$

For $m = 1, r = 1$

$$\sum_{i=1}^{4}\sum_{k=i=1}^{4} D_{ik} + D_{ki} = (D_{12} + D_{21} + D_{13} + D_{31} + D_{14} + D_{41}) + (D_{23} + D_{32} + D_{24} + D_{42}) + (D_{34} + D_{43})$$

$$= (0 + 0.83 + 2.25 + 4.5 + 0 + 1.33) + (0 + 0 + 0 + 1.3) + (0 + 0)$$

$$= 10.21$$

Therefore total D_{in} for $m = 1, 2, 3, 4, 5, 6$

$$D_{in} = 10.21 + 14 + 10.5 + 8 + 11.75 = 54.46$$

$$D_{out} = \sum_{m=1}^{M-1}\sum_{i=r_1}^{s-1}\sum_{n=m+1}^{M}\sum_{k=r_2}^{s}(D_{ik} + D_{ki})$$

For $m = 1, i = 1, n = 2, 3, 4, 5, 6$

Sum$_{11}$ = $\sum_{k=6,8,9,10} D_{1k} + D_{k1} + \sum_{k=7,10,11} D_{1k} + D_{k1} + \sum_{k=5} D_{1k} + D_{k1} + \sum_{k=12}^{15} D_{1k} + D_{k1} + \sum_{k=16}^{23} D_{1k} + D_{k1}$

$= 0 + 0 + 1 + 0 + 0 = 1$

Sum$_{12}$ =

$\sum_{k=6,8,9,10} D_{2k} + D_{k2} + \sum_{k=7,10,11} D_{2k} + D_{k2} + \sum_{k=5} D_{2k} + D_{k2} + \sum_{k=12}^{15} D_{2k} + D_{k2} + \sum_{k=16}^{23} D_{2k} + D_{k2}$

$= 0 + 0 + 1.25 + (0.83*4) + (0.83*4) = 11.21$

Sum$_{13}$ = $\sum_{k=6,8,9,10} D_{3k} + D_{k3} + \sum_{k=7,10,11} D_{3k} + D_{k3} + \sum_{k=5} D_{3k} + D_{k3} \sum_{k=12}^{15} D_{3k} + D_{k3} + \sum_{k=16}^{23} D_{3k} + D_{k3}$

$= 0 + 0 + 0 + 0 + 0 = 0$

Sum$_{14}$ =

$\sum_{k=6,8,9,10} D_{4k} + D_{k4} + \sum_{k=7,10,11} D_{4k} + D_{k4} + \sum_{k=5} D_{5k} + D_{k5} + \sum_{k=12}^{15} D_{4k} + D_{k4} + \sum_{k=16}^{23} D_{4k} + D_{k4}$

$= 0 + 2 + 0 + 0 + 0 = 2$

Sum$_1$ for module 1 = 1 + 11.21 + 0 + 2 = 14.21

Total D_{out} = 14.21 + 12 + 0 + 0 + 0 + 0 = 26.21

Therefore,

Relative Modularity for the Modularized Workflow – INLP results

= 54.46/ (54.46 + 26.21) = 0.68

2. Relative Modularity for the Modularized Workflow – Heuristic results

Total D_{in} = 14.56 + 7.5 + 0 + 5.75 + 4.5 + 0 = 32.31

Total D_{out} = 25.7 + 10.5 + 2.5 + 7.5 + 2 + 0 = 48.2

Therefore,

Relative Modularity for the Modularized Workflow – Heuristic results
= 32.31/ (32.31 + 48.2) = 0.40

The modularized workflow obtained from the INLP results seems to be more modular than that obtained with the heuristic, although both results scored low, 0.68 and 0.40, where the possible range of values is 0 to 1. These relative modularity values are useful to compare workflow design options and to guide the redesign process for creating flexible workflows. Further optimization of task assignment could be done to increase modularity by comparing the different modules and evaluating for task re-assignment.

CHAPTER 6

6 CONCLUSIONS AND FURTHER WORK

Modular design is an important form of strategic flexibility [12], that is flexible designs allow a company to respond to changing markets by creating product variants derived from different combinations of existing or new modular components. Similarly for modular workflow designs organizations can more efficiently reconfigure their workflows to meet environmental changes.

This effort sought to define and measure workflow modularity. Three main objectives were proposed:

1. Design a modular workflow process
2. Develop a mathematical model (OR model) to define modules of modular workflow process, including identifying potential configurations
3. Develop a measure for workflow process modularity

An integer nonlinear programming model (INLP), called a modularization model was used to cluster atomic tasks of a workflow into loosely coupled modules based on task interdependencies and flow time. The model proved useful for modularizing small workflows, however for larger workflows it is inefficient as a result of the increased complexity in obtaining an optimal solution.

A heuristic that focuses on balancing tasks to modules based on interdependence strength and the organizational unit responsible for the tasks provides a solution for larger workflows. The heuristic proved to provide adequate results in less time.

In summary the following are the contributions of this work:

4. A modular workflow design considering flow time and flexibility, two of the most important performance measures in workflow design
5. An INLP optimization model for designing modular workflows that can be adopted for small processes
6. A heuristic for creating modules that can be adopted for larger processes

From a research perspective, this effort has added to the continued work on business process redesign and specifically on workflow analysis and design, through the application of modularity concepts.

Additional research can be directed in the following:

1. Addition of decision variables and constraints to INLP model to handle parallel processing of tasks.
2. Accommodation of additional measures of workflow performance such as cost.

REFERENCES

1. Marshall, R. and P.G. Leaney. *A Systems Engineering Approach to Product Modularity*. in *Proceedings of the Institution of Mechanical Engineers Part B*. 1999.

2. Mikkola, J.H. and O. Gassmann, *Managing Modularity of Product Architectures: Toward an Integrated Theory.* IEEE Transactions of Engineering Management, 2003. 50(2): p. 204 - 218.

3. Tully, *The Modular Corporation.* Fortune, 1993: p. 52-56.

4. Baldwin, C.Y. and K.B. Clark, *Design Rules*. 2000: Published by MIT Press, Cambridge. 63-92.

5. Kusiak, A., *Integrated Product and Process Design: A Modularity Perspective.* Journal of Engineering Design, 2002. 13(3): p. 223-231.

6. Gershenson, J.K., G.J. Prasad, and S. Allamneni, *Modular Product Design: A life-cycle view.* Journal of Integrated Design and Process Science, 1999. 3(4): p. 13 - 26.

7. Gershenson, J.K. and G.J. Prasad, *Modularity in product design for manufacturing.* International Journal of Agile manufacturing, 1997. 1(1): p. 99-109.

8. Ethiraj, S.K. and D. Levinthal, *Modularity and Innovation in Complex Systems.* INFORMS, Management Science, 2004. 50(2): p. 159-173.

9. Newcomb. *Implications of modularity in product design for the life cycle*. in *ASME Design Engineering*. 1998. Irvine, CCA.

10. Pahl, G. and W. Beitz, *Engineering Design: A systematic Approach*. 1988: Springer-Verlag.

11. Ulrich, K. and K. Tung. *Fundamentals of product modularity*. in *ASME Design Engineering*. 1991. Miami, Fl.

12. Sanchez, R. and J.T. Mahoney, *Modularity, Flexibility, and Knowledge Management in Product and Organization Design.* Strategic Management Journal, 1996. 17(Winter Special Issue): p. 63-76.

13. Wheelwright, S.C. and K.B. Clarke, *Revolutionizing Product Development: Quantum Leaps in Speed, Efficiency and Quality*. 1992: Free Press, New York.

14. Langlois, R.N. and P.L. Robertson, *Networks and Innovation in modular system: Lessons from the microprocessor and stereo component industry.* Research Policy, 1992. 21(4): p. 297-313.

15. Nevins, J.L. and D.E. Whitney, *Concurrent Design of Products & Processes: A strategy fro next generation in manufacturing.* 1989: McGraw-Hill.

16. Tully, S., *The Modular Corporation.* Fortune, 1993: p. 106-114.

17. Sanderson, S.W. and V. Uzumeri, *Strategies for new product development and renewal: Design-based incrementalization.* 1990.

18. Sanchez, R. and D. Sudharshan, *Real-Time Market research: Learning by doing in development of new products.* Marketing Intelligence and Planning, 1993. 11(7): p. 29-38.

19. Cusumano, M.A., *Japan's Software Factories: A challenge to U.S. management.* 1991: Oxford University Press.

20. Woolsey, J.P., in *777 Air Transport World.* 1994. p. 22-31.

21. Ceiton, *Web Workflow PPS.* 2004.

22. Kamrani, A.K. and S.e.M. Salhieh, *Product Design for Modularity.* 2000: Kluwmer Academic Publishers. 19 - 48.

23. Baldwin, C.Y. and K.B. Clark, *Managing in an Age of Modularity.* Harvard Business Review, 1997. 75(5): p. 84 -93.

24. Yigit, A.S., A.G. Ulsoy, and A. Allahverdi, *Optimizing Modular Design for Reconfigurable Manufacturing.* Journal of Intelligent manufacturing, 2003. 13: p. 309 - 316.

25. Kamrani, A.K. and R. Gonalez, *A Genetic Algorithm-based solution methodology for modular design.* Journal of Intelligent manufacturing, 2003. 14: p. 599-616.

26. Gershenson, J.K., G.J. Prasad, and Y. Zhang, *Product Modularity: Measures and Design Methods.* Journal of Engineering Design, 2004. 15(1): p. 33-51.

27. Thormann, U.W. and M.L. Brandeau, *Optimal Commonality in Component Design.* Operations Research, INFORMS, 1997. 48(1): p. 001-019.

28. Worren, N., K. Moore, and P. Cardona, *Modularity, strategic flexibility, and firm performance: A study of the Home Appliance Industry.* Strategic Management Journal, 2002. 23: p. 1123 -1140.

29. Ali and Gonalez, *Genetic algorithm model for developing similar components in designing complex products.* 2003.

30. Fisher, M., K. Ramdas, and K. Ulrich, *Component Sharing in the management of product variety.* Management Science, 1999. 45(3): p. 297-315.

31. Ulrich, K.T. and D. Ellison, *Holistic customer requirements & design-select decision.* Management Science, 1999. 45(5): p. 641-658.

32. Ulrich, K., et al., *Including the value of time in design-for-manufacturing decision making.* Management Science, 1993. 39(4): p. 429-447.

33. Evans, D.H., *Modular Design.* Operations Research, 1963. 11: p. 637-647.

34. Hammer and Champy, *Re-engineering the Corporation.* 1993.

35. Basu, A. and R.W. Blanning, *A Formal Approach to Workflow Analysis.* Information Systems Research - INFORMS, 2000. 11(1): p. 17 - 36.

36. Reijers, A.H., *Design and Control of Workflow Processes.* 2003: Springer.

37. Schal, T., *Workflow Management Systems for Process Organizations.* 1996: Springer-Verlag Berlin Heidelberg.

38. Government, C.f.T.i., *An Introduction to Workflow Management Systems.* 1997, University of Albany /SUNY. p. 1-16.

39. Coalition, W.M., *Workflow facility specification.* 1997. p. 9 -21.

40. Browning, T.R., *Designing System Development Projects for Organizational Integration.* 1999. Regular Paper: p. 217 -223.

41. Malone, T.W. and K. Crowston, *The Interdisciplinary Study of Coordination.* ACM Computing Surveys, 1994. 26: p. 87-119.

42. Victor, B. and R.S. Blackburn, *Interdependence: An alternative conceptualization.* Academy of Management, 1987. 12: p. 486-498.

43. Bertsimas, D. and J.N. Tsitsiklis, *Introduction to Linear Optimization.* 1997: MIT.

44. Hillier, F.S. and G.J. Lieberman, *Introduction to Operations Research.* Seventh ed. 2001: Tata McGraw-Hill. 332-339.

45. Winston, W.L., *Introduction to Mathematical Programming: Applications and Algorithms.* 1995: Duxbury Press.

46. Wybo and Goodhue, *Using Interdependence as a predictor of data standards: Theoricatical and Measurement issues.* Information and Management, 1995. 29: p. 317-329.

47. Steward, D.V., *The Design Structure System: A method for managing the design of complex systems.* IEEE Transactions of Engineering Management, 1981 b. 28(3): p. 71-74.

48. Steward, D.V., *Sytems Analysis and Management:Structure, strategy and design.* 1981a: Princeton, NJ.

49. Giachetti, R.E., *Understanding Interdependence in enterprises: A model and measurement formalism.* 2004: p. 22.

50. Chen, S.-J. and L. Lin, *Decompostion of interdependent task group for concurrent engineering.* Computer and Industrial Engineering, 2003. 44(2003): p. 435-459.

51. Browning, T., *Applying the design structure matrix to system decomposition and integration problems:A review and new directions.* IEEE Transactions of Engineering Management, 2001. 48(3): p. 292-306.

52. Chung-Yu, J., S.-J. Chen, and L. Li, *A structured approach to measuring functional dependency and sequencingof coupled tasks in engineering design.* Computer and Industrial Engineering, 2003.

53. Kusiak, A. and C.-C. Huang, *Modularity in Design of Products and Systems.* IEEE Transactions on Systems, Man, and Cybernetics, 1998. 28(1).

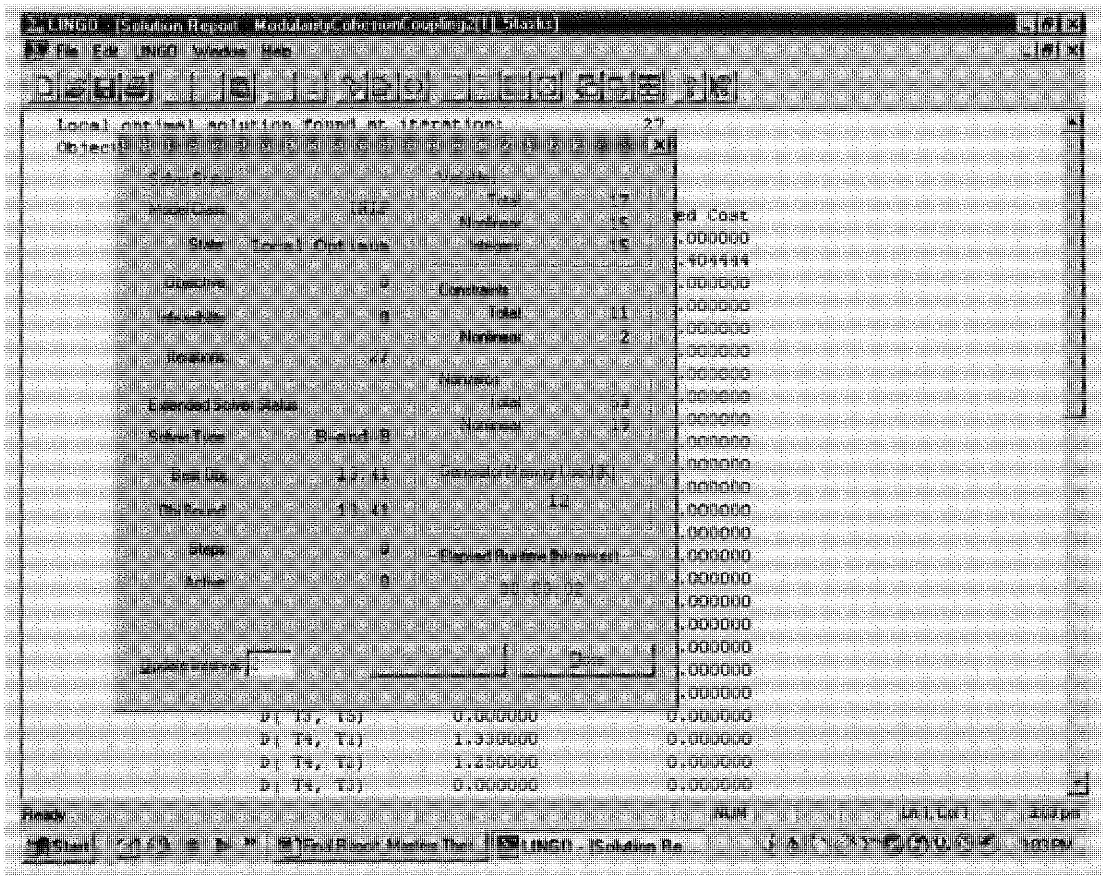
54. Yassine, A.A. and D.M. Sharman, *Characterizing Complex Product Architecture.* Regular Paper, 2003.

55. Kusiak, A., T. Larson, and J. Wang, *Reengineering of Design and Manufacturing Processes.* Journal of Mechanical Design, 1994. 26(3): p. 521-536.

APPENDICES

APPENDIX 1

A. Lingo Results for 5 tasks DSM



```
LINGO - [Solution Report - ModularityCohesionCoupling2[1]_5tasks]
File  Edit  LINGO  Window  Help

  Local optimal solution found at iteration:              27
  Object
                                                              ed Cost
                                                              .000000
                                                              .404444
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                                                              .000000
                       D( T3, T5)          0.000000           0.000000
                       D( T4, T1)          1.330000           0.000000
                       D( T4, T2)          1.250000           0.000000
                       D( T4, T3)          0.000000           0.000000
```

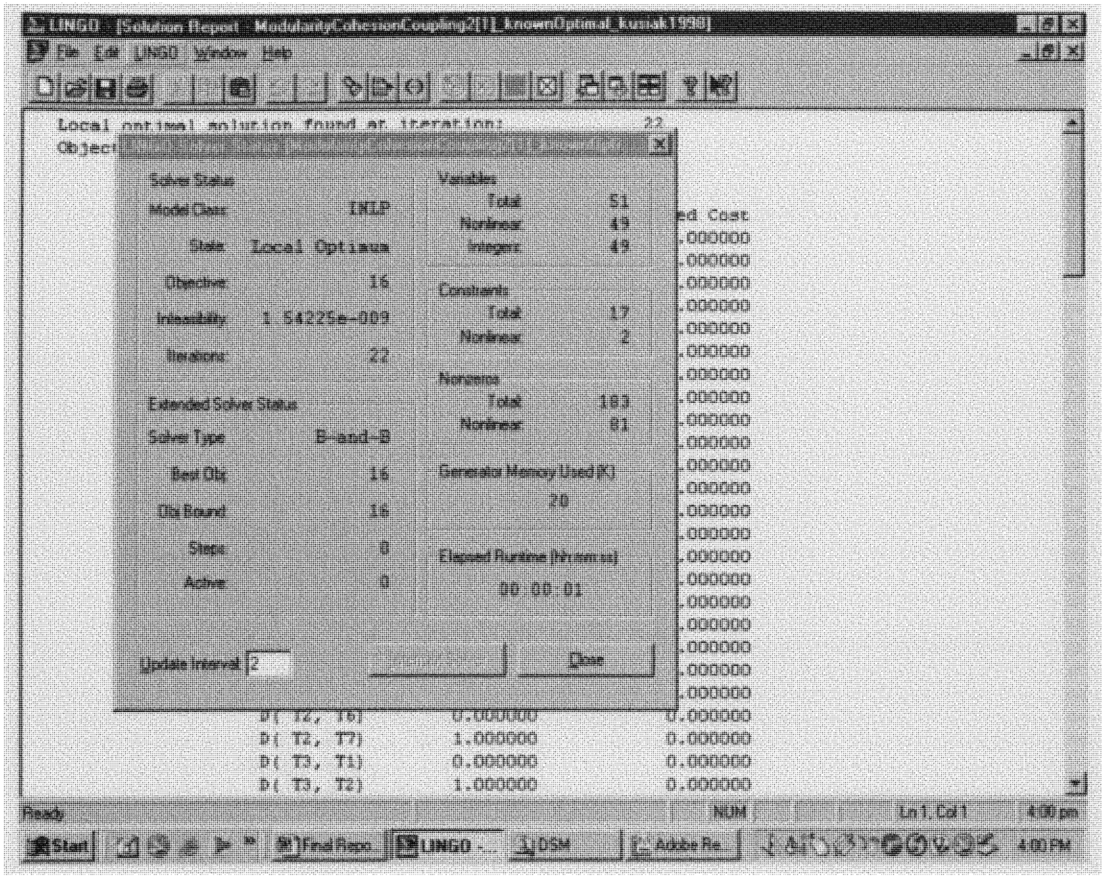
```
Solver Status                        Variables
  Model Class:           INLP            Total:          17
                                         Nonlinear:      15
  State:        Local Optimum            Integers:       15

  Objective:                0         Constraints
                                         Total:          11
  Infeasibility:            0            Nonlinear:       2

  Iterations:              27         Nonzeros
                                         Total:          53
Extended Solver Status                   Nonlinear:      19
  Solver Type:         B-and-B
                                      Generator Memory Used (K)
  Best Obj:              13.41                 12
  Obj Bound:             13.41

  Steps:                    0         Elapsed Runtime (hh:mm:ss)
  Active:                   0                00:00:02

Update interval: 2                                    Close

Ready                                          NUM          Ln 1, Col 1     3:03 pm
```

A1: Modularity Matrix

	2	5	4	1	3
2	2				
5	1	5			
4	1		4		
1	2			1	
3	1			5	3

Compare to optimal solution of Module 1 = {2, 5, 4), Module 2 = {1, 3}

B. Lingo Results for 7 tasks DSM



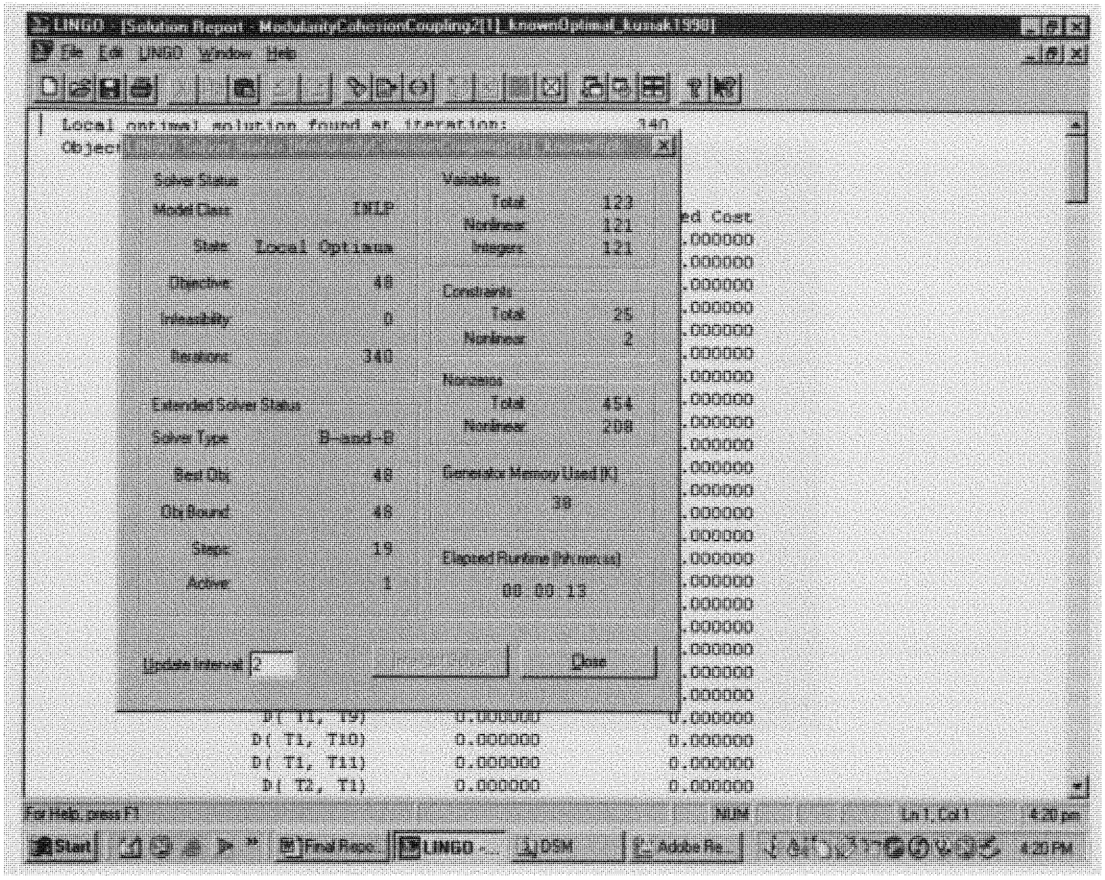
```
LINGO - [Solution Report - ModularityCohesionCoupling2[1]_knownOptimal_kusiak1998]

Local optimal solution found at iteration:          22

Solver Status                      Variables
  Model Class          INLP          Total          51
  State      Local Optimum           Nonlinear      49
  Objective              16          Integers       49
  Infeasibility 1.54225e-009       Constraints
  Iterations             22          Total          17
                                     Nonlinear       2
Extended Solver Status             Nonzeros
  Solver Type        B-and-B         Total         183
  Best Obj               16          Nonlinear      81
  Obj Bound              16        Generator Memory Used (K)
  Steps                   0                 20
  Active                  0        Elapsed Runtime (hh:mm:ss)
                                            00:00:01
Update Interval 2                              Close

                  D( T2, T6)        0.000000        0.000000
                  D( T2, T7)        1.000000        0.000000
                  D( T3, T1)        0.000000        0.000000
                  D( T3, T2)        1.000000        0.000000
```

B1: Modularity Matrix

	A	E	F	B	C	D	G
A	A	1	1				
E		E	1			1	
F	1	1	F				
B				B			
C				1	C	1	1
D		1		1	1	D	1
G				1	1	1	G

Compare to optimal solution of Module 1 = {A, E, F), Module 2 = {D, B, C, G}

C. Lingo Results for 11 tasks DSM



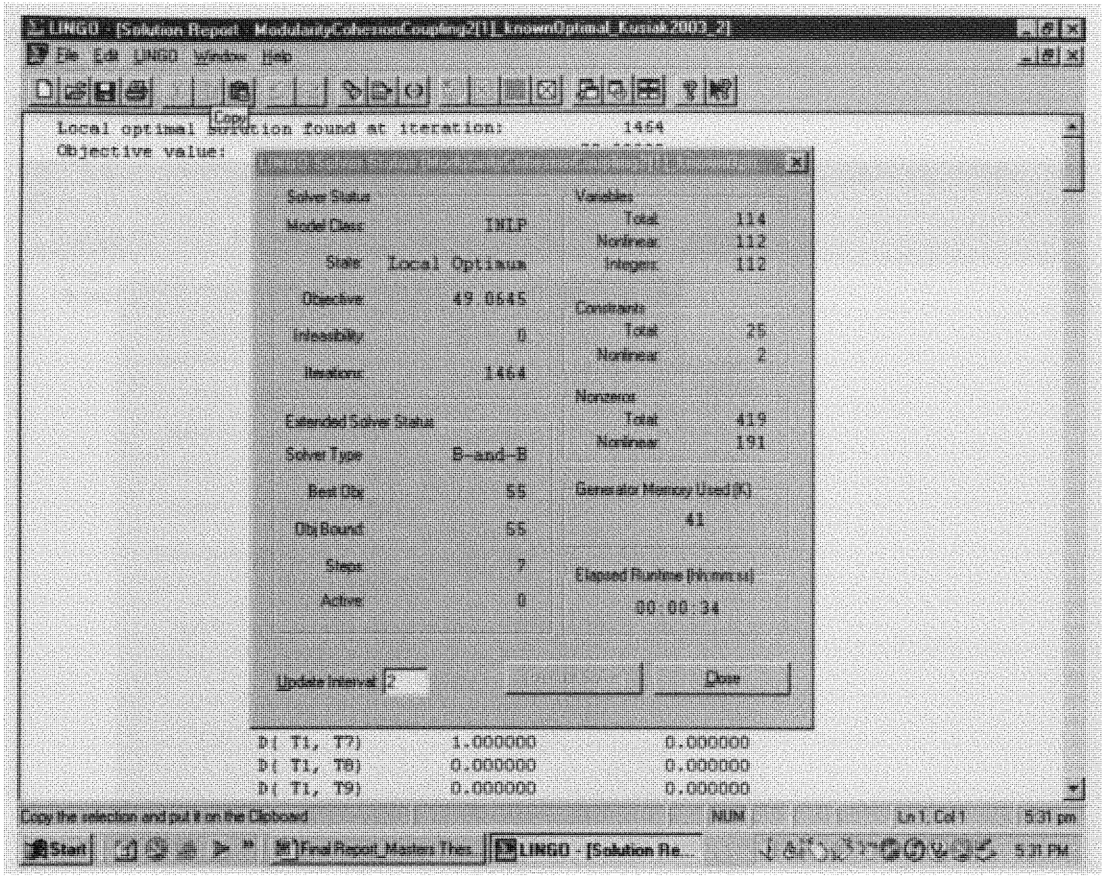
```
LINGO - [Solution Report - ModularityCohesionCoupling2[1]_knownOptimal_kusiak1990]
File  Edit  LINGO  Window  Help

  Local optimal solution found at iteration:           340
  Object

  Solver Status                        Variables
  Model Class            INLP              Total          123
                                           Nonlinear      121
        State   Local Optimum              Integers       121
    Objective              48
                                       Constraints
  Infeasibility             0              Total           25
                                           Nonlinear        2
   Iterations             340
                                       Nonzeros
  Extended Solver Status                   Total          454
  Solver Type          B-and-B             Nonlinear      208
     Best Obj              48          Generator Memory Used (K)
    Obj Bound              48                    38
        Steps              19          Elapsed Runtime (hh:mm:ss)
       Active               1                00:00:13

  Update Interval 2                                  Close

                                                             ed Cost
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                                                            .000000
                    D( T1, T9)         0.000000            0.000000
                    D( T1, T10)        0.000000            0.000000
                    D( T1, T11)        0.000000            0.000000
                    D( T2, T1)         0.000000            0.000000
For Help, press F1                          NUM              Ln 1, Col 1    4:20 pm
Start   Final Repo...   LINGO -...   DSM   Adobe Re...   4:20 PM
```

C1: Modularity Matrix

	2	7	8	11	3	5	1	4	6	9	10
2	2		5								
7		7									
8		5	8	5							
11				11		5					
3					3	5	1	1	2	2	2
5				5	5	5					
1					1		1				
4					1			4			
6					1				6		
9					1					9	
10					1						10

Compare to optimal solution of Module 1 = {7, 2, 8), Module 2 = {11, 5, 3}

D. Lingo Results for 14 tasks DSM



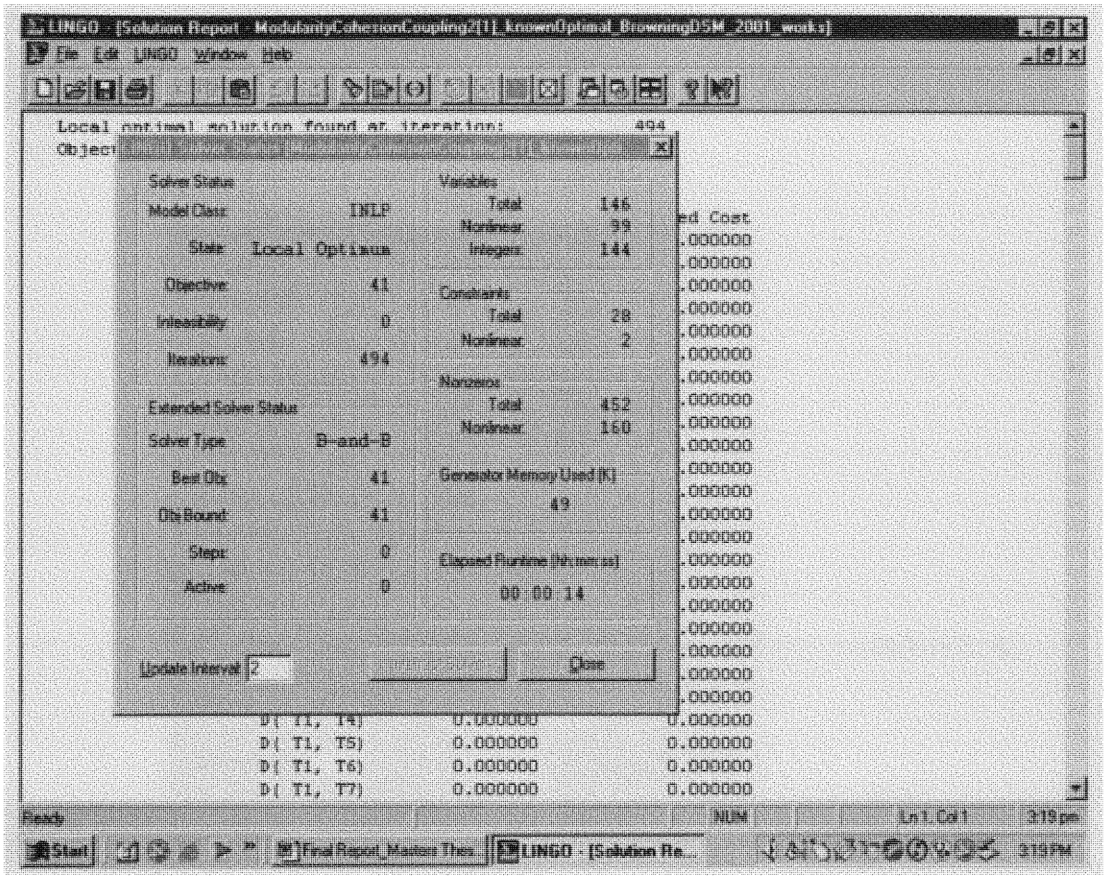
```
LINGO - [Solution Report - ModularityCohesionCoupling2[1]_knownOptimal_Kusiak2003_2]
File  Edit  LINGO  Window  Help

  Local optimal solution found at iteration:          1464
  Objective value:

Solver Status                          Variables
  Model Class:            INLP            Total:        114
        State:   Local Optimum         Nonlinear:       112
    Objective:         49.0645           Integer:       112
  Infeasibility:             0         Constraints
   Iterations:            1464             Total:        25
                                       Nonlinear:         2
Extended Solver Status                 Nonzeros
  Solver Type:          B-and-B            Total:       419
     Best Obj:               55        Nonlinear:       191
    Obj Bound:               55        Generator Memory Used (K)
        Steps:                7                41
       Active:                0        Elapsed Runtime (hh:mm:ss)
                                           00:00:34
Update Interval: 2                                        Close

                D( T1, T7)        1.000000          0.000000
                D( T1, T8)        0.000000          0.000000
                D( T1, T9)        0.000000          0.000000

Copy the selection and put it on the Clipboard          NUM        Ln 1, Col 1      5:31 pm
```

D1: Modularity Matrix

	2	4	5	8	9	6	11	12	13	14	1	3	7	10
2	2		2					1						
4		4	3		2									
5	2	3	5	3	1									
8			3	8										
9		2	1		9			1						
6						6	3		3					
11						3	11	3						
12	1						3	12	3				1	
13						3		3	13	2				
14									2	14	2		1	
1											1		1	
3												3		2
7				1				1		1	1	1	7	1
10														10

Compare to optimal solution of Module 1 = {4, 5, 8}, Module 2 = {13, 11, 6, 12}, Module 3 = {10, 3}, Module 4 = {7, 14, 1}.

E. Lingo Results for 16 tasks DSM



```
LINGO - [Solution Report - ModularityCohesionCoupling2(1)_knownOptimal_BrowningDSM_2001_works]
File  Edit  LINGO  Window  Help

Local optimal solution found at iteration:          494
Object

Solver Status                               Variables
  Model Class              INLP                 Total        146
                                                Nonlinear     99
  State           Local Optimum                 Integers     144
  Objective                  41             Constraints
  Infeasibility               0                 Total         28
  Iterations                494                 Nonlinear      2
                                            Nonzeros
Extended Solver Status                          Total        452
  Solver Type             B-and-B               Nonlinear    160
  Best Obj                   41             Generator Memory Used (K)
  Obj Bound                  41                          49
  Steps                       0             Elapsed Runtime (hh:mm:ss)
  Active                      0                      00:00:14

Update Interval 2                                          Close

                                                   ed Cost
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                                                   .000000
                    D( T1, T4)      0.000000       0.000000
                    D( T1, T5)      0.000000       0.000000
                    D( T1, T6)      0.000000       0.000000
                    D( T1, T7)      0.000000       0.000000
Ready                                   NUM          Ln 1, Col 1     3:19 pm
Start   Final Report_Masters Thes...   LINGO - [Solution Re...       3:19 PM
```

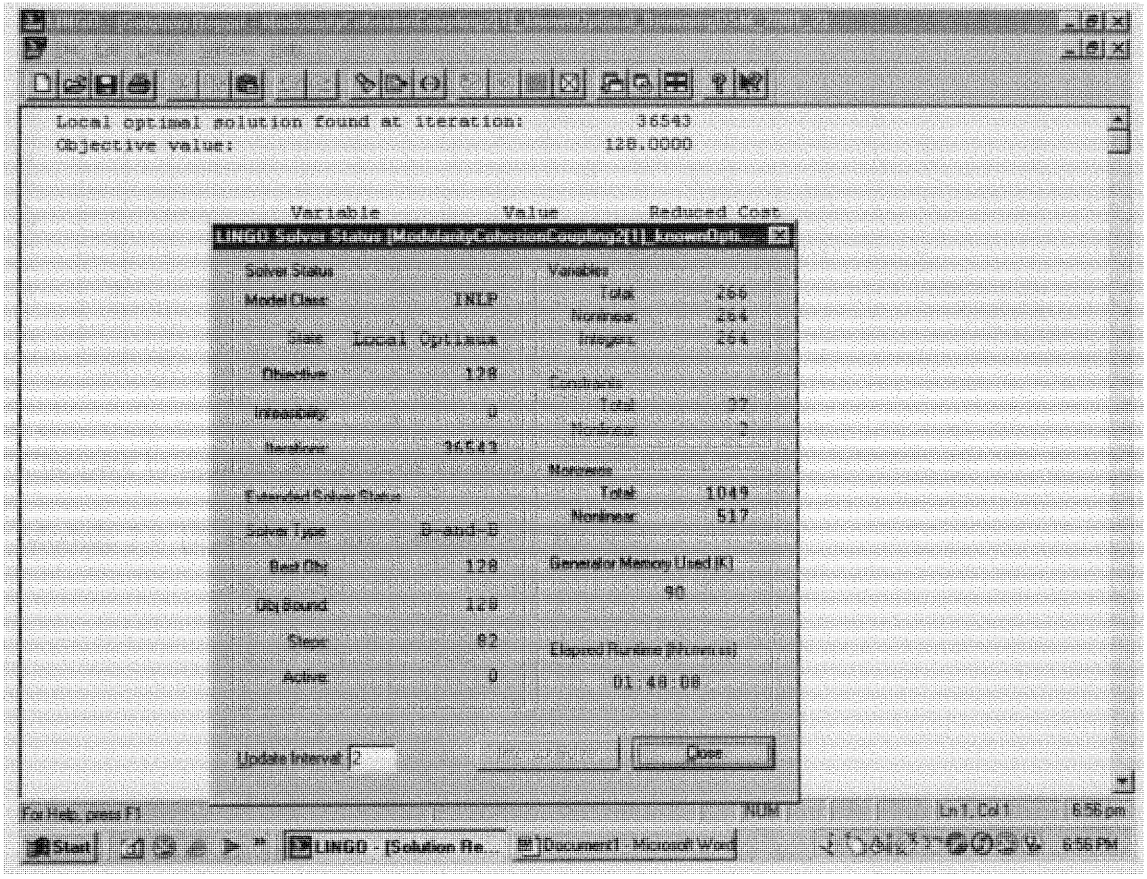
E1: Modularity Matrix

	A	B	E	F	I	H	C	P	O	G	D	J	K	L	M	N
A	A	2														
B	2	B	2													
E		2	E	2		2										
F			2	F	2	2										
I				2	I	2										
H			2	2	2	H										
C							C	2								
P						2	2	P		2						
O								2	O							
G								2		G						
D											D					
J												J				
K													K			
L														L		
M															M	
N																N

Compare to optimal solution of Module 1 = {A, B), Module 2 = {E, F, I},

Module 3 = {H, G, P, O, G}

F. Lingo Results for 22 tasks DSM



```
Local optimal solution found at iteration:        36543
Objective value:                                  128.0000

              Variable              Value         Reduced Cost
```

LINGO Solver Status [ModularityCohesionCoupling2[1]_knownOpti...

Solver Status		Variables	
Model Class:	INLP	Total:	266
State:	Local Optimum	Nonlinear:	264
Objective:	128	Integers:	264
Infeasibility:	0	Constraints	
Iterations:	36543	Total:	37
		Nonlinear:	2
Extended Solver Status		Nonzeros	
Solver Type:	B-and-B	Total:	1049
Best Obj:	128	Nonlinear:	517
Obj Bound:	128	Generator Memory Used (K)	90
Steps:	82	Elapsed Runtime (hh:mm:ss)	01:40:08
Active:	0		

Update Interval: 2

F1: Modularity Matrix

	F	E	G	D	I	A	C	B	K	J	N	Q	R	O	L	M	H	P	S	T	U	V
F	F	2	2	2	2	2													1			1
E	2	E		2	1	1																1
G	2		G			1																
D	1	1		D	1	2												1				
I	1			1	I	2	1															
A	2			2	2	A	2	2		2									2		2	
C					1	2	C	2														1
B				1		2	2	B	2	2				1	2		2	1	1			2
K						1		2	K	2	1	2	2	1	2	2	2	2	1	1		2
J				1	1	2	1	2	1	J							2					1
N									2		N	2		2			2					
Q									2	1	2	Q	2			1	1	1		2		1
R												2	R					1		1		
O									1		1			O	2		2			1		1
L								2	2					2	L	2		1	1	1		1
M								1	2						2	M				2		1
H	1					2		2	2	2	1	2		2	1	1	H	2				1
P									2								1	P			1	1
S	2		2			2	2	2	1						2		1	2	S	2	2	2
T	1								1	2		1	2	1		1		2	2	T	2	1
U			1		1	2	1								2			1	2	2	U	2
V	2	1	1	2	2	2	1	2	2	1		1		1	1	1	1	2	2	1	2	V

Compare to optimal solution of Module 1 = {F, G, E, D, I, A), Module 2 = {C, B, K, J}, Module 3 = {P, N, Q, R, O}, Module 4 = {L, M}