Attempt to Integrate System Dynamics and UML in Business Process Modeling

Liang-Cheng Chang
Department of Information Management
National Sun Yat-Sen University, Kaohsiung 804, Taiwan
886-7-5252000 ext. 4760
d9142809@student.nsysu.edu.tw

Yi-Ming Tu
Department of Information Management
National Sun Yat-Sen University, Kaohsiung 804, Taiwan
886-7-5252000 ext. 4717
ymtu@mis.nsysu.edu.tw

Abstract

The integration of information systems and business process will affect competitive advantages of firms. In order to develop information system, modeling of business process is a fundamental work of system analysis and design. System dynamics is useful to solve non-linear, complex, time delay and feedback problems of business processes. However it still belongs to a special field of modeling language because it can't be integrated well with information systems in organizations. The purpose of this paper is to integrate system dynamics with UML and thus they can be developed synchronously during information systems implementation in enterprise. For this reason, integrated development process and system architecture with system dynamics and UML have also been proposed in this paper.

Keyword: System Dynamics, UML, Object-Orientation, System Analysis and Design

1. Introduction

Every company has a number of business processes which is some kinds of activities within the organization where people or machines work together to achieve desired outcome or their business objectives (Ould, 1995). For activities in business processes, Hammer (1996) classified activities into three types, Value-adding work, Non-value-adding work, and Waste. Porter (1985) divided activities into support activities and primary activities. Support activities include firm infrastructure,

human resource management, technology development, procurement and Primary activities include inbound logistics, operation, outbound logistics, marketing and sales, and service. Porter (1985) argued that firms can raise competitive advantages from primary value added activities. Therefore, the design of business process would affect competitive advantages of firms.

Harmer (1990) proposed a concept of Business Process Reengineering that is the radical redesign of business processes to cut non-value-adding work and waste. The main concept of reengineering is to integrate business process with information technologies. Thus traditional paper-intensive process must be broken, and be redesigned in the center of information technologies. In order to make business process to adapt to information technologies, many business process modeling methodologies have been proposed.

Rumbaugh (1999) pointed out that model is a representation in a certain medium of something in the same or another medium and model captures the important aspects of the thing being modeled from a certain point of view and simplifies or omits the rest. A model of a software system is made in a modeling language and it has both semantics and notation and can take various forms that include both pictures and text.

Recently, some technologies to model real world and to develop software systems have been presented, Object-Orientation, Function-Orientation, State-Orientation and Formal (Peters, 2000). Object-Orientation is the most popular method which could make software product to be reusable, portable, and interoperability (Schach, 2002). For enterprises, if they model business process by object method, they can adjust their system more flexibility.

UML (Unified Modeling Language) is the standard object-oriented modeling language proposed by Booch, Jacobson and Rambaugh. During the software development process, software engineers can communicate to each other through UML which includes 10 diagrams by static and dynamic views. In this paper, we apply UML to model business process because it has been the standard modeling language of OMG (Object Management Group).

Besides previous methodologies, System dynamics is another methodology to model business process. Senger (1990) identified business process as a circle of causality that describes a feedback loop of cause and effect, and it could be depicted by system perspective. Sequentially, Sterman (2000) argued that system dynamics can be applied to model business process and to solve complex problems included Constantly Changing, Tightly Couple, Governed by Feedback, Non-Linear, History

Dependent, Self-Organizing and Counter-intuitive. As we known, most practical business processes have above characteristics. Therefore, system dynamics seems to be more powerful than normal software modeling methodologies especially in dynamic and non-linear processes and most of business problems all have these features.

In spite of system dynamics has a lot of advantages to become a model language of business information systems, it has still been confined in a specific filed and doesn't belong to a popular software modeling language. Tu (2003) pointed out the limitation of traditional simulation software of system dynamics. Firstly, the isolated interface of traditional software doesn't support the demand in decision of the enterprise today to update current data from database. Secondly, when traditional software simulates each policy or scenario, it must reset the data and execute the simulation process every time. Besides, most traditional software products were programs of single computer and they couldn't be integrated with business information systems through networks. If companies want to apply system dynamics to be a powerful tool to solve their problems, they must integrate system dynamics with their information systems. It would be better to develop information systems and system dynamics synchronously in the beginning.

Based on previous reasons, the purpose of this paper is attempt to integrate system dynamics with UML to model business process and also to propose an integrated information system development process by system dynamics and UML. The second paragraph of this paper is to compare components of object-oriented methodology and system dynamics. The third paragraph is to transform causal diagram and stock-flow diagram to UML diagram by the case of market-growth in order to show that stock flow diagram and UML diagrams can be exchange each other. Fourth paragraph is to propose an integrated development process of system dynamics and UML. Conclusion and suggestion will be presented in last paragraph.

2. Comparison of Object-Orientation and System Dynamics

The concept of object-orientation is a revolution of software development. Object-Oriented Analysis and Design (OOAD) makes software development time shorten and system maintain easily. By the definition of OMG, object is a combination of a state and a set of methods that explicitly embodies an abstraction characterized by the behavior of the relevant requests (OMG, 1991).

A main feature of object-orientation is to consider both data and actions to be equally important and to combine data and action in Class that is a template for several objects and presents internal structure of object (Jacobson, 1992;

Sommerville, 2004). A class is composed by Attribute and Behavior (Jacobson, 1992). Data in class are called Attributes which mean characteristics in order to describe state of class. By the perspective of information system, attributes are data or information gathered from database or inputted from users. Actions in class are called Behavior or Operation that means the action or method to stimulate or to change the state of class. Here, we apply the item "operation" in this paper. By dynamic view of class, attributes may be affected by operation.

Components of system dynamics are Sector, Level, Rate and Auxiliary. Level or Stock characterizes the state of the system and provides the basis for actions. It also provides system with inertia and memory (Sterman, 2000). Rate or Flow is the change of level with time. By the definition, rate is the action to change the state of the level with time.

Class can be regarded as the sub-system of information systems from the definition of class in object-orientation and sector can also be regarded as sub-system in system dynamics. Therefore, we can find that class and sector have similar characteristic which is the sub-system of the whole system. They can also be denoted by mathematic description of Set { }.

The characteristics of Level and Attribute are also similar and they can both present the state of system. By the definition, level and attribute can be denoted by a variable Y which means a state of system. Besides, auxiliary can also express the property of system or it can be set as an initial condition or other variables affected by environment. So an auxiliary can also be regarded as an attribute because they both present data or information. Auxiliaries can be denoted as a constant C, other variable X, function of level Y, function of other variable X or function of time.

The characteristic of Rate and Operation is similar and it presents the action which changes the state of system. By the definition, the mathematic symbol of Rate or Operation is differentiation of level Y with time.

In summary, the comparison of features in system dynamics, object-orientation and their mathematic description are summarized in Table 1.

Table 1 Comparison of system dynamics and object-orientation

System Dynamics Component	**Object-Orientation Concept**	**Mathematic Description**
Sector	Class	Set{ }
Level (Stock)	Attribute	Y
Rate (Flow)	Operation (Behavior)	$\frac{dY}{dt}$
Auxiliary	Attribute	C、X、F(Y)、F(X)、F(t)

3. Transformation of Stock Flow Diagram and UML Diagram

3.1 Introduction of SD diagrams and UML diagrams

The diagrams of system dynamics (SD) include casual diagram and stock flow diagram. During the process of system dynamics, the problem of system is portrayed by casual diagram and next step is to model and to simulate the problem by stock flow diagram.

Unified Modeling Language (UML) is a standard language in order to model software system and it was integrated by Rational from object-oriented modeling tools proposed by Booch, Rambaugh and Jacobson. The purpose of UML is specifying, constructing and documenting object-orientation system by visualizing (Booch et al, 1999). There are 9 diagrams of UML and 5 perspectives of system architecture presented in Table 2

Table 2 UML diagrams and system perspectives (Reed, 2002)

Perspective	**UML Diagram**	**Application**
Use	Use Case Diagram	Events of users, system analyzers, or system engineers to use the system
Design	Class Diagram, Object Diagram	To design the functions of system or the service to users of system
Process	Sequencing Diagram, Collaboration Diagram,	To present the synchronic sequence or process of system

	State Diagram, Activity Diagram	
Deployment	Component Diagram	To present the components or files of system
Implementation	Deployment Diagram	To present network infrastructure or hardware of system implementation

The connection of system dynamics diagrams and UML have been exhibited during past years. Tignor (2003) compared the features of activity diagram of UML and stock flow diagram of system dynamics and then built a stock flow diagram based on UML activity diagram. The main contribution of Tignor is to transform activity diagram to stock flow diagram.

Sequentially, Tignor (2004) applied the case of British Telecom Intelligent Network from UML diagrams to causal diagram and stock flow diagram. Tignor also found the possibility of applying system dynamics to the problems of modeling information systems.

Previous researches showed the possibility of transformation from UML diagrams to SD diagrams. However, the study of transformation from SD diagrams to UML diagrams is still scarce. The purpose of this paper is to apply a model of system dynamics to transform from SD diagrams to UML diagram.

3.2 Simplified Market Growth Model

The part of the model “Market Growth” built by Forrester in 1975 would be applied in this paper. Original model included four causal loops, marketing effort, market perception, production capacity utilization, and capital investment which could be regarded as four business processes. In this paper, there are two loops drown out from original model and be depicted as Sales and Orders. The casual diagram of simplified market growth model is shown in Figure 1. In the loop of sales, hiring salesman will increase number of salesman and more salesmen will earn more orders booked. Then more orders will get more budgets of sales and more budgets can let company hire more salesmen. We could find it is a positive loop for sales loop. In another loop of orders, more orders booked will take more backlogs and more backlogs will bring more delivery delay. There is a time delay between delivery delay and sales effectiveness and a negative effect from delivery delay to sales effectiveness. At last, better sales effectiveness will earn more orders booked. We could find it is a negative loop for orders loop.

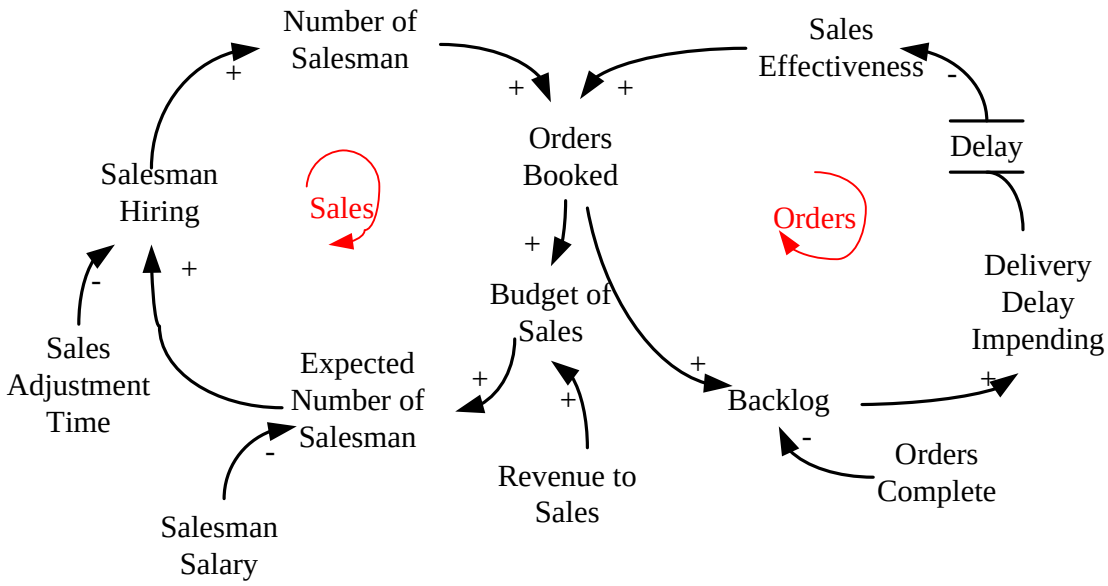


Figure 1 Causal diagram of Simplified Market Growth Model

Then, the stock flow diagram is built in Figure 2. The loop "Sales" has been depicted as sector "Sales" and another loop "Orders" has been depicted as sector "Orders". In the sector of salesman, the number of salesman has been depicted as the level of salesman, and hiring salesman has been depicted as the rate of salesman hiring. Others items are auxiliaries. In the sector of orders, backlog has been depicted as the level of backlog and delivery delay is another level. Order entered and order complete are the rate of backlog and changes in DDR (Delivery Delay Recognized) is the rate of delivery delay recognized. Auxiliaries in sector of orders are also shown in Figure 2.

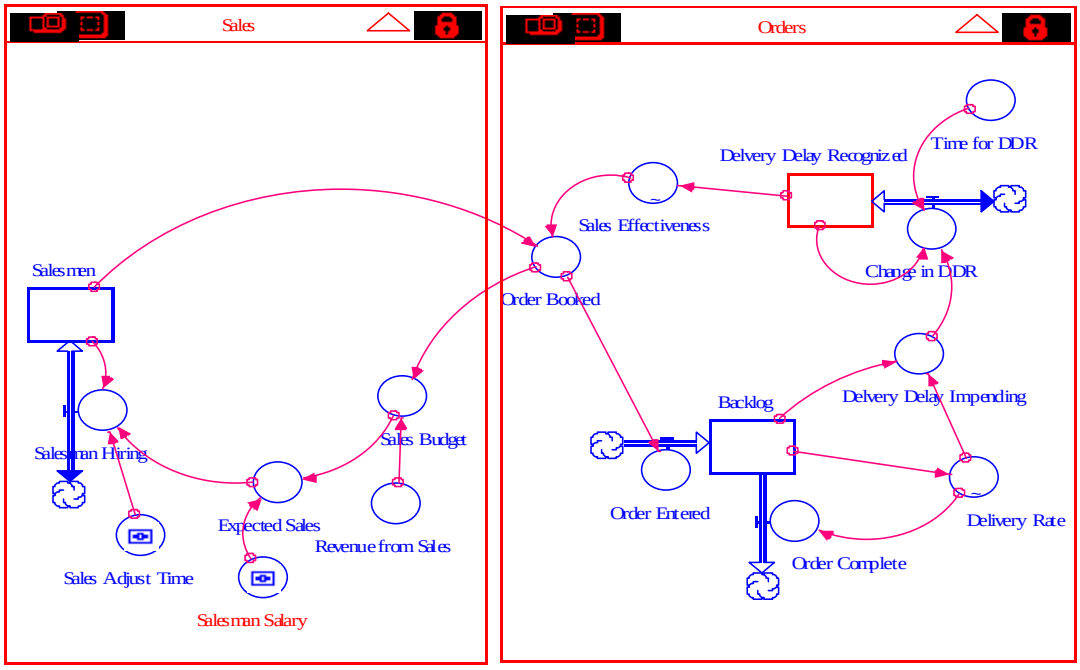


Figure 2 Stock-Flow diagram of Simplified Market Growth Model

The simulation result of this simplified market growth model is presented in Figure 3. The quantities of Sales, Backlog, and Delivery delay recognized are presented an S-shape growth, and sales effectiveness depresses and falls during the time. Two loops of sales and orders could be regarded as business process and processes of sales and orders affect each other and it becomes the system for an archetype of limitation of growth proposed by Senge (1990). It's hard to recognize this kind of system complexity in business process by traditional information systems, but system dynamics can. Therefore, implement system dynamics into traditional information system would be inevitable task.

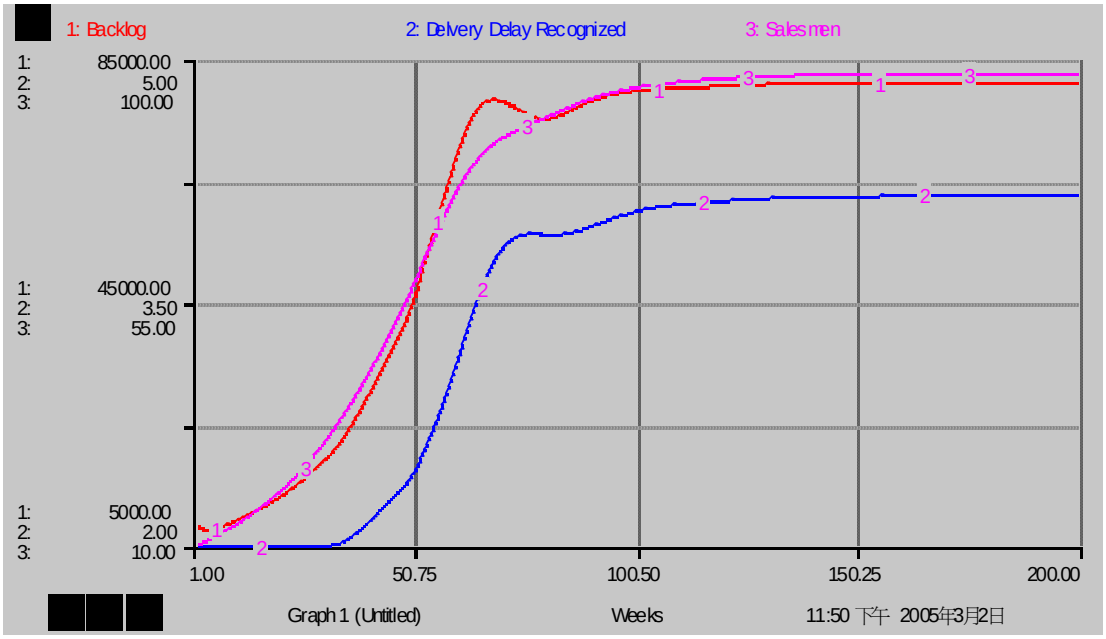


Figure 3 Simulation Result of Simplified Market Growth Model

3.3 Modeling Simplified Market Growth by UML

We will build UML diagrams based on SD diagram that is an opposite way to previous researches. Firstly, by the definition of Table1, components of level, rate and auxiliary in stock diagrams should be transformed to class, attribute and operation in object-orientation concept. The results of SD diagram components corresponding with Object-Orientation Components are shown in Table 3

Table 3 SD Components Corresponding with Object-Orientation Components

Components of Object-Orientation	**Components of System Dynamics**	**Component of Stock-Flow Diagram in Figure 2**	
Class	**Sector**	Sales	Orders

Attribute	**Auxiliary**	Sales Adjust Time	Order Booklog
		Expected Sales	Sales Effectiveness
		Salesman Salary	Time for DDR
		Revenue from Sales	Delivery Delay Impending
		Sales Budget	Delivery Rate
Attribute	**Level**	Salesman	Delivery Delay Recognized
			Backlog
Operation	**Rate**	Salesman Hiring	Order Entered
			Order Complete
			Change in DDR

Secondly, according to the content of Table 3, class diagram based on stock-flow diagram of simplified market growth could be depicted as Figure 4. Two sectors of stock-flow diagrams would be transformed to be two classes, sales and orders in class diagrams and attributes and operations are also be shown as below.

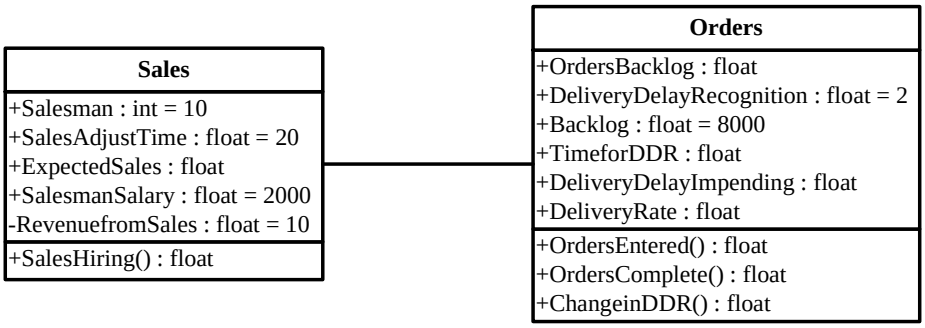


Figure 4 Class Diagram of Simplified Market Growth Model

Thirdly, based on stock-flow diagram in Figure 2 and class diagram in Figure 4, sequencing diagram could be depicted as Figure 5. System users input data and get result from system through user interface. The simplified market growth model can be built as a marketing information system. According to Figure 2, rate of salesman hiring is affected by sales adjust time and expected sales, so user input the data of these attribute to system, and then get information of salesman from system. The information of salesman would be transferred from class sales to class orders. After

user input related information into system, user can get information about backlog and delivery delay recognized from information system.

During the process from stock flow diagram to sequencing, we found some problems about transformation. First, stock flow diagram is synchronous, but sequencing diagram is asynchronous. It is difficult to decide which attributes or operations should be list first in sequencing based on stock flow diagram. Second, sequencing diagram didn't have the concept of time delay, but stock flow did. It's difficult to describe time delay in sequencing diagram.

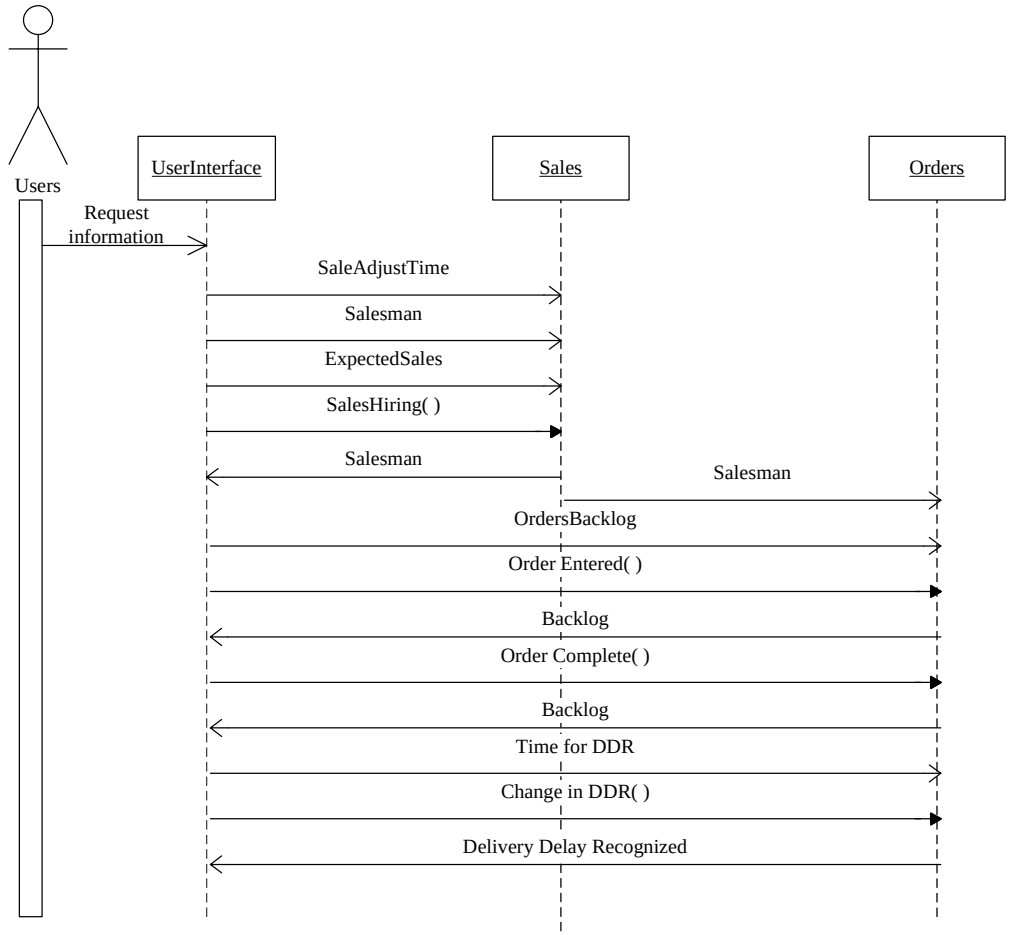


Figure 5 Sequencing Diagram of Simplified Market Growth Model

Fourthly, based on sequencing diagram, activity diagram can be depicted as Figure 6 which shows different activity states of system. It starts from hiring salesman and this activity state will affect state of salesman. Then, state of salesman will affect state of order booked which branches to budget and to order entered. On state of backlog, the flow branches again to delivery rate and delivery delay impending. At state of delivery delay impending, the flow goes to change in DDR, delivery delay recognized, and sales effectiveness. After state of sales effectiveness,

the flow goes back to order booked. Other flows are also presented in Figure 6.

Comparing with sequencing diagram, activity diagram is more similar with stock flow diagram because they are built by concept of flow whatever data flow or activity flow. Therefore we consider that activity diagram of UML would be most similar with stock flow diagram of SD. It's corresponding with previous researches.

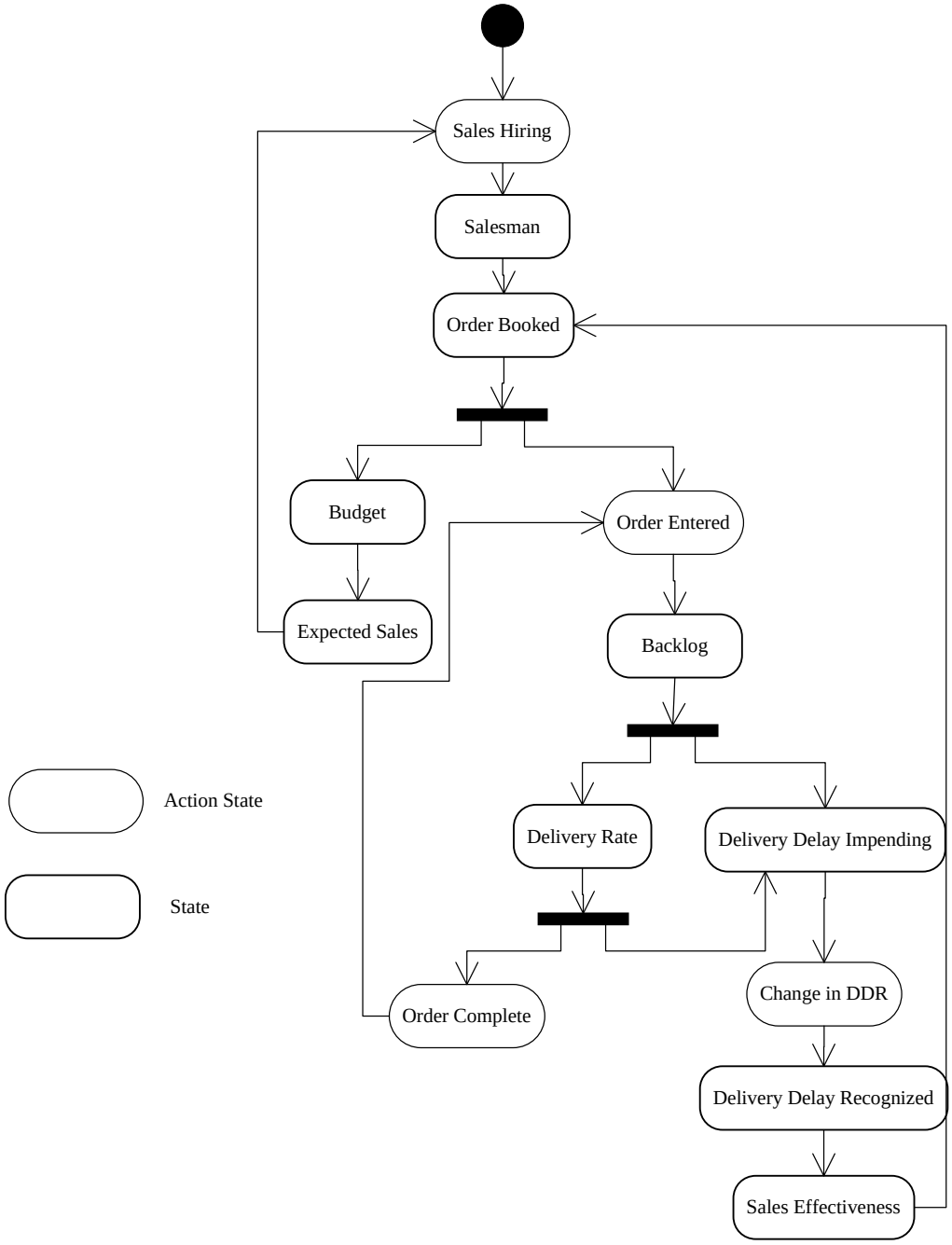


Figure 6 Activity Diagram of Simplified Market Growth Model

Finally, programming code of DYNAMO for system dynamics and JAVA for UML could also be presented in Table 4. Programming codes in Tables is only to compare the structure of these two different programming languages, and some

detail mathematic functions of program would be ignored. Actually, calculation of differential equations in system dynamics model needs to be calculated by numerical methods like Euler's method or Newton's method. If these numerical methods are written by programming language, it will be large programs with many lines of codes. Therefore, for non-linear problems which need to solve complex partial differential equations, DYNAMO would be easier to program than normal programming languages like JAVA, C++ or Visual BASIC. If system dynamics can be integrated with information systems and DYNAMO can be embodied in programming languages, many complex non-linear problems can easily to be solved.

Table 4 Simplified Market Growth Model Programmed by DYNAMO

Programming Codes by DYNAMO	Programming Codes by JAVA
Sector: Sales **L Salesman.K=Sales.J+(DT)(SalesmanHiring.JK)** **N Salesman=10** **A SalesBudget=(OrderBooked)** **A OrderBooked=(Salesman.K)(RevenuefromSales)** **C RevenuefromSales=10** **A ExpectedSales=SalesBudget.K/SalemanSalary** **C SalesmanSalary=2000** **R SalesmanHiring.KL=(ExpectedSales.K-Salesman.K)/SaleAdjustTime** **C SalesAdjustTime=20** **Sector: Orders** **A OrderBooked.K=(Salesman.K)(SalesEffectiveness.K)** **R OrderEntered.KL=OrderBooked.K** **L Backlog.K=Backlog.J+(DT)(OrderEntered.JK=OrderComplete.JK)** **N Backlog=8000** **A DeliveryRate.K=TABLE(DR,Backlog.K,0,100000,20000)** **T TDR=0/10000/16000/18500/19500/20000** **R OrderComplete=DeliveryRate.K** **A DeliveryDleayImpending.K=Backlog.K/DeliveryDelayRecognized.K** **R ChangeinDDR.KL=(DeliveryDelayImpending.K)/TimeforDDR** **C TimeforDDR=6** **L DeliveryDelayRecognized.K=DeleiveyDelayRecognized.J+(DT)(ChangeinDDR.JK)** **N DeliveryDelayRecognized=2** **A SalesEffectiveness.K=TABLE(TSE,DeliveryDelayRecognized.K,0,6,1)** **T TSE=400/390/350/290/210/150/100**	**Class Sales {** **Public float Salesman=10;** **Public float SaleAdjustTime;** **Public float ExpectedSales;** **Public float SalemanSalary=2000;** **Public float RevenuefromSales;** **Public SalesmanHiring(ExpectedSales, Salesman, SaleAdjustTime)** **{ FUNCTION(ExpectedSales,Salesman,SaleAdjustTime);** **Return Salesman; }** **}** **Class Orders {** **Public float OrdersBacklog;** **Public float DeliveryDelayRecognized=2;** **Public float Backlog=8000;** **Public float TimeforDDR;** **Public float DeliveryDelayImpending;** **Public float DeliveryRate;** **Public float SalesEffectiveness;** **Public OrdersEntered(OrderBooked, Baclog) {** **FUNCTION (OrderBooked,Backlog);** **Return Backlog;}** **Public OrdersComplete(DeliveryRate,Backlog){** **FUNCTION(DeliveryRate,Backlog);** **Return Backlog;}** **Public ChangeinDDR(DeliveryDelayImpending, TimeforDDR, DeliveryDelayRecognized){ FUNCTION (DeliveryDelayImpending, TimeforDDR, DeliveryDelayRecognized);** **Return DeliveryDelayRecognized;}** **}**

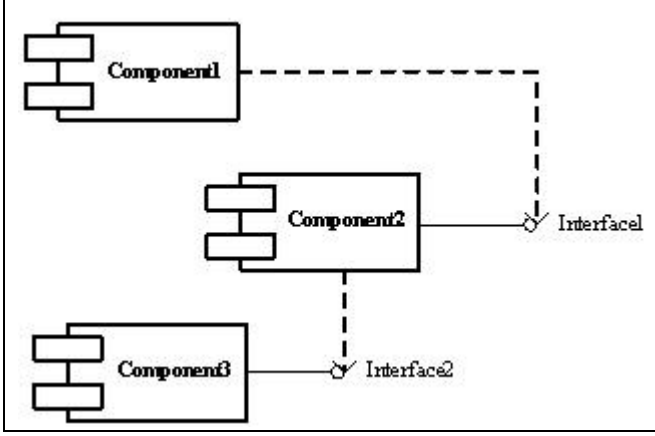
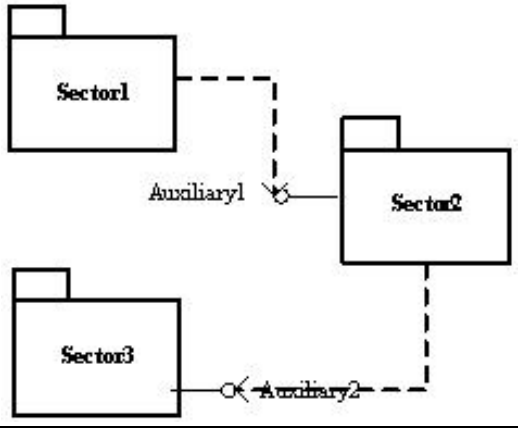
4. Integrated Software Systems and Development Process for UML and SD

4.1 Composition of Re-usable System Dynamics Models

Current large software is usually constructed by components which is a tested, special-purpose software unit. A software component is useful, adaptable, portable, and interoperable (Peters, 2000). So many large information systems are combined by components which make systems be reused and be maintained easily.

Some software tools of system dynamics also provide functions reusable models. Powersim Studio argued their software products support for object-orientation and reusable models. Users can define a library of individual models for reuse (Hauke, 2001). However, it is still not software components which can be integrated with existed information systems. In this paper, the concept of reusable software components for system dynamics model would be proposed. Component diagram of UML is shown in the right side of Table 6. Components in component diagrams can mean an executable file, a data file, a picture or a movie. Components are operated or communicated through interface. In the left side of Table 6, software components of system dynamics model is be presented. Many existed models can be encapsulated as sectors and every sector can be connected by auxiliaries. Then, a large model can be composed by existed sectors.

Table 6 Component Diagrams of UML and SD

Component Diagram of UML	Sectors composition of System Dynamics
	

For example, in the basis of Table 6, an integrated new business dynamics model can be constructed as Figure 6. There were some existed system models which built for growth and learning, internal process, customers, and finance. Someday, a company wants to build a Balanced Scorecard (Kaplan, 1996) and they can

composed a new system dynamics model by existed models which can be regarded as sectors and connected them by auxiliaries. By an integrated development process with SD and UML which will be discussed later, they can develop information systems and system dynamics models synchronically and develop large SD models by methodologies of object-orientation.

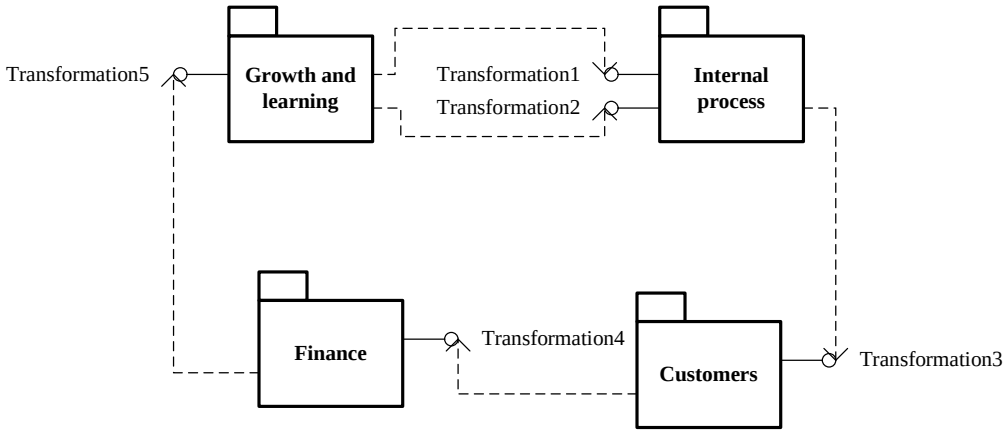


Figure 6 An Integrated System Dynamics Model

4.2 Integrated Development Process

For software development process, three principles identified in IEEE 1074-1995 standard are:

1. Requirements, to decide what a system must do.
2. Design, to determine how a system computes.
3. Implementation, to produce source code, documentation, and tests

These three principles are fundamental processes for software development. After that, many software development processes are also proposed which included waterfall software process model, incremental life cycle model, spiral life cycle model, prototyping model, extreme programming, and Rational Unified Process. Every development process has its advantage and disadvantage and software engineers decide which one to choose depends on the characteristics of projects.

Rational Unified Process is the development process designed for UML, and it includes initial planning, business modeling, requirements, analysis & design, implementation, test, and deployment (Kruchten, 1998). In this paper, UML is the methodology to model business process and therefore Rational Unified Process is also the development process be chosen.

In System Dynamics modeling process, Sterman (2000) listed them as Problem Articulation (Boundary selection), Dynamic Hypothesis, Formulation, Testing, and

Policy Formulation & Evaluation were the iterative modeling process presented as Figure 7.

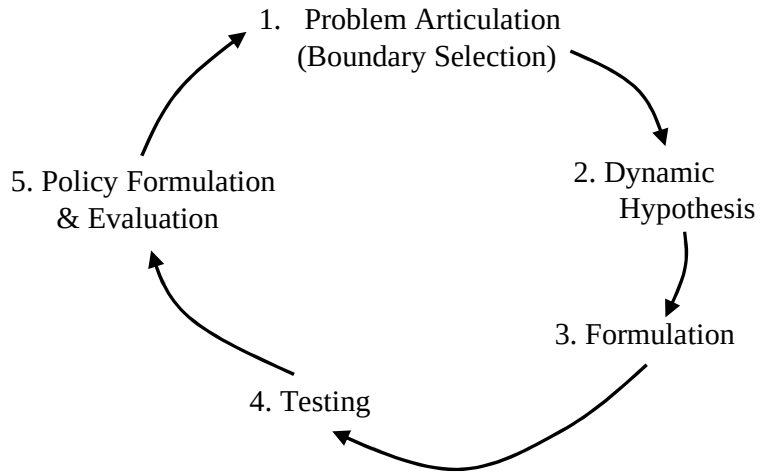


Figure 7 Iterative System Dynamics Modeling Process (Sterman, 2000)

The phase of problem articulation is to decide what the problem is and horizon of time. In dynamic hypothesis, develop maps of causal structure based on current theories. Then, formulate a simulation model. In phase of testing, test robustness under extreme condition and sensitivity. At last, policy design and evaluation is scenario specification (Sterman, 2000).

After compared with SD development process and Rational Unified Process, we found that there were many similar phase between these two processes. An integrated system development process with system dynamics and UML is presented as Table 7.

Table 7 Integrated Development Process

Rational Unified Process	**System Dynamic Modeling Process**	**Diagrams of SD and UML**
Initial Planning	Problem Articulation	Use Case
Business Modeling	Dynamic Hypothesis	Use Case
Requirements		Use Case, Casual
Analysis	Formulation	Class, Object, Collaborative, Causal
Design		Component, Sequencing, Activity, State, Stock-Flow

Implementation	Testing	Deployment, Stock-Flow
Test		Stock-Flow
Deployment	Policy Formulation & Evaluation	Stock-Flow

In this integrated development process, UML and System Dynamics models are analysis and design synchronously. Through simulation by system dynamics, we can evaluate final result of information system before it be deployed and correct it iteratively.

For test and deployment phase of software development process, there is no suitable diagrams in the process. If we want to test software, we need to complete the whole program or at least, we need to have a prototype. Nevertheless, stock-flow diagram can become the role of test and deployment. System dynamics can also do scenario testing for information systems.

4.3 Integrate Information Systems

Base on the integrated development process, the integrated information system proposed by Tu (2003) would be implemented. Traditional business information systems included Transaction Process System (TPS), Management Information System (MIS) and Decision Support System (DSS). After we combined UML and SD, the result simulated by system dynamics can be the data of decision support system. System dynamics software can also get data from database to overcome shortages of current system dynamics soft wares.

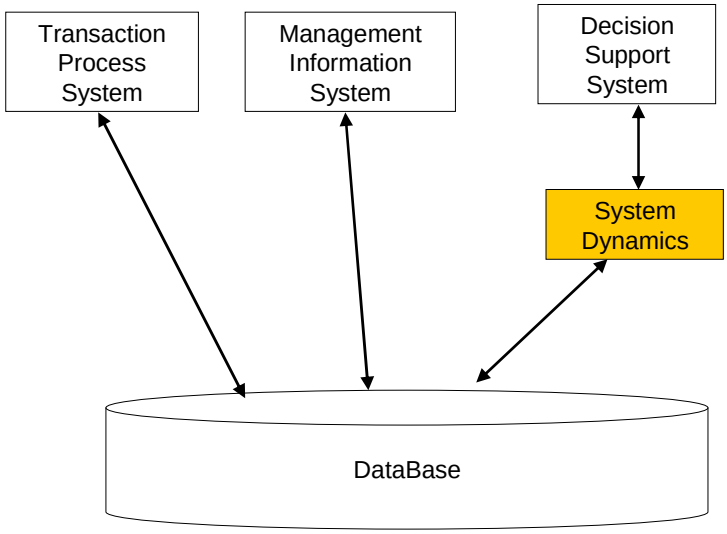


Figure 9 Integrated Information Systems (Tu, 2003)

By the perspective of system structure, traditional information system would be

composed by data-bases, application programs, and user interface. An integrated structure adds system dynamics into the position of application programs which can develop like Table 4 and Table 5.

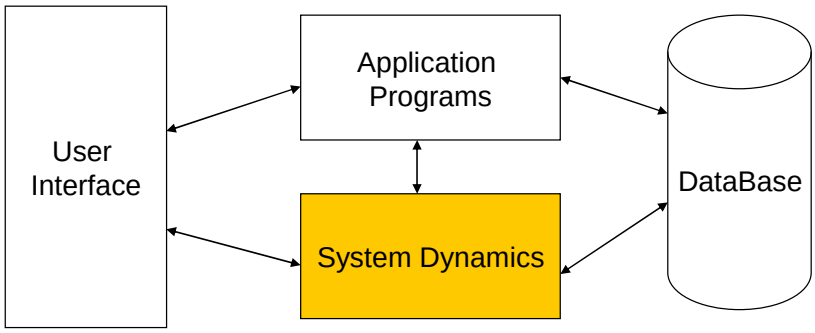


Figure 10 Integrated Software Structure

5. Conclusion and Suggestion

System dynamics needs to be integrated with business information systems, and it can be really applied popularly in the future. UML is the most popular modeling language to describe business process and we compare it with system dynamics. After compared with system dynamics and UML, we found that system dynamics can solve the problems of time delay and system complexity which traditional information systems can't. Therefore, concept of integrated information system with UML and SD has been proposed.

In order to build the integrated information system, system dynamics diagrams and UML diagrams can be drowned and program codes can also be transformed synchronically. By the concept of components, a new and large system dynamics model can be constructed by existed models which regard as sectors. Therefore system dynamics model can also be developed by methodologies of software engineering. Design and implementation of physical integrated information will be next topic in future research.

Reference

Booch, G. Rumbaugh, J. and Jacobson, I. 1999. *The Unified Modeling Language User Guide*. Addison Wesley Longman, Inc.

Forrester, J. 1975. Market Growth as Influenced by Capital Investment. *Collected papers of Jay Forrester.* Portland, OR: Productivity Press, 11-132

Hammer, M. 1990. Reengineering Work: Don't Automate, Obliterate. *Harvard Business Review 68(4)*: 104-113

Hammer, M. 1996. *Beyind Reengineering*. HaperBusiness

Hauke, U. and Berende, K. 2001. *Powersim's Business Modeling and Simulation*

Tools Are Built Right In to SAP SEM. http://www.powersim.com/common/pdf/sap-powersim.pdf

Jacobson, I. 1992. *Object-Oriented Software Engineering: A Use Case Driven Approach*. Addison Wesley Publishing Company

Kaplon, RS. and Norton, DP. 1996. *The Balanced Scorecard: Translating Strategy into Action.* Harvard Business School Publishing Corporation.

Kruchten, P. 1998. *The Rational Unified Process: An Introduction*. Addison Wesley Longman, Inc.,

Marshall, M. 2000. *Enterprise Modeling with UML*. Addison Wesley Longman, Inc.

Ould, MA. 1995. *Business Processes*. John Willey & Sons

Object Management Group. 1991. *The Common Object Request Broker: Architecture and Specification*.

Peters, JF and Pedrycz, W. 2000. *Software Engineering: An Engineering Approach*. John Wiley & Sons, Inc.

Porter, MJ. 1985. *Competitive Advantage*. Free Press: New York

Reed, Jr. PR. 2002. *Developing Applications with JAVA and UML*. Pearson Education Inc.

Rumbaugh, J., Jacobson, I. and Booch, G. 1999. *The Unified Modeling Language Reference Manual*. Addison Wesley Longman Inc.

Schach, SR. 2002. *Object-Oriented and Classical Software Engineering*. 5th ed. McGraw-Hill Companies, Inc.

Senge, PM. 1990. *The Fifth Discipline: The Art and Practice of the Learning Organization*. Doubleday Currency.

Sommerville, I. 2004. Software Engineering. 7th ed. Addison Wesley

Sterman, J D. 2000. *Business Dynamics*. McGraw-Hill Companies Inc.

Tignor, W. 2003. Stock and Flow, and Unified Modeling Language Relationships. *Proceedings of 21th International Conference of the System Dynamics Society*.

Tignor, W. 2004. System Engineering and System Dynamics Models. *Proceedings of 22nd International Conference of the System Dynamics Society*

Tu, YM and Wu, TF. 2003. A Study of Connect Dynamic Data Source to Improve the Simulate Technique for the System Dynamics. *Proceedings of 21th International Conference of the System Dynamics Society*.