To Main Proceedings Document

A Simulation Methodology Unifying System Dynamics and Business Objects

Andreas Gregoriades & Dr Vasilis Karakostas
Dept. of Computation, UMIST University, Manchester M60 1QD, UK
email: andreas@ist.co.umist.ac.uk

Abstract

In order to encourage the participation of business people in the modelling phase and to abstract business concepts from their simulation complexities the proposed framework combines system dynamics with business objects. The approach eliminates the isolated modelling for simulation and operation and enforces a common business model that accomplishes both. The unification consequently assists in concurrent evolution of the simulation and operational model. Since the simulation model is directly linked to the operational model information collected during systems operation is directly communicated to the simulation model for processing and vice versa. In addition, the integration provides an intuitive interface to the simulation engine based on real world concept.

For the unification of business object and system dynamics paradigms, the Powersim constructor and Visual Basic 6.0 are employed. The former is used to construct the system dynamics models and the latter to develop the infrastructure for the business process and business objects. A number of system dynamics co-models are distributed in equal number of business objects each one related to the business aspects that the object represents. Business objects run the various co-models simultaneously and exchange information concerning the simulation through the business object infrastructure. The business rules incorporated in each business object act as navigators for the encapsulated simulation model and the business object itself.

Introduction

System dynamics is a simulation technique that is based on the principles of systems thinking. The underline philosophy concentrates on the interconnections of systems elements as a mean for a holistic representation of reality. Organisations are considered as systems composed of business entities that are governed by business processes. Business objects on the other hand are the main building blocks of those business processes. This paper introduces a simulation methodology that unifies system dynamics and business objects and explains the advantages of such a unification.

Systems thinking & Business Objects

Systems thinking, is more than anything else, a mindset for understanding how things work. It is a perspective for going beyond events, to looking for patterns of behaviour, to seeking underlying systemic interrelationships which are responsible for the patterns of behaviour and events.

System dynamics is a methodology that takes the information about a system's structure and formalises it into a computer related model using the concepts of system thinking (Cover, 1996). The word “dynamic” implies constant change, and indeed, that is what systems do - change over time. The basic building blocks of system dynamics are stock, flows and feedback loops. Stocks are accumulators and flows define the rate that stock are filled or emptied.

Object modelling is a special approach to the construction of models of complex systems, in which a complex system, consisting of a large number of occurrences, is regarded as a set of objects. The relations between these occurrences are seen as associations between these objects; their properties are attributes of the objects. In addition, the occurrence that affects another when a certain event takes place is described as communication between objects.

Business objects provide a natural and ideal way to model the business. The reason for that is that they form meaningful units because they are drawn directly from the real-world objects in business. Unlike programming language objects, business objects depict on business concepts such as customers and sales orders rather than the operational aspects of the application that they represent.

Overview of the Architecture

The approach focuses on the principle of business objects which abstract from real world concepts whilst avoiding unnecessary detail. On the other hand, simulation models are focused on business entities and the interaction between them. Since business entities are common in both methodologies there is no need to have duplicate models to represent the same business process.

In this methodology, business objects form a higher level abstraction of a system dynamics model. A group of system dynamics entities, that represent a clear business concept are encapsulated in a corresponding business object that reflect the same business concept as the system dynamics entities. Business Objects at a higher level form feedback loops in the same way that entities in system dynamics do. Elements from the embedded system dynamics model (called a co-model) send or receive parameters from other elements that reside in different business objects (i.e. from other co-models) in order to complete loops that may be hidden inside business objects (figure 1).

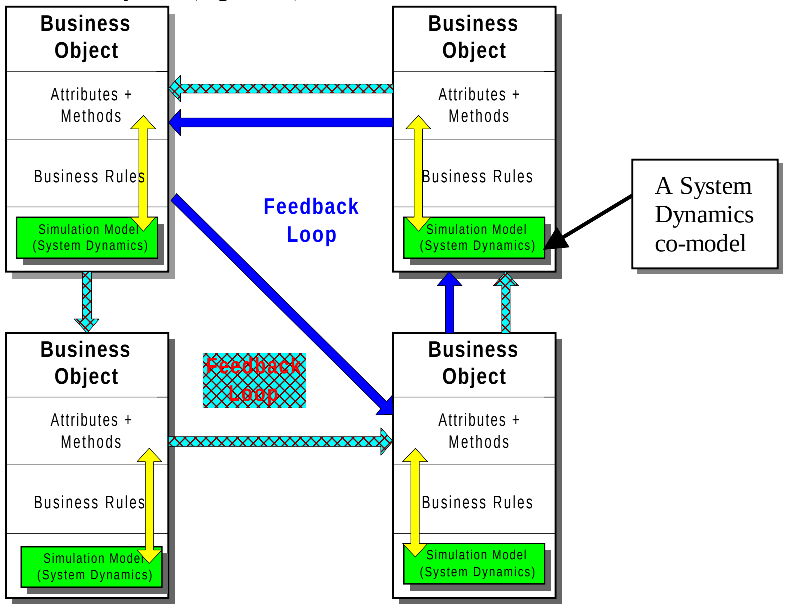


Figure 1 Business objects forming feedback loops with each other while at the same time communicate the state of co-operating business object's co-models to their own simulation model.

As mentioned before, each business object encapsulates a segment of the entire system dynamics model. Business rules encapsulated in business objects supply the simulation model with information that is translated into its equivalent numerical format prior to being processed by the simulation model. Simulation is dependent on the business objects that provide the parameters to the corresponding fragment of the simulation model. In turn each business object depends on other business objects which provide it with information concerning the results of their embedded co-models. All system dynamics co-models are executed simultaneously by their corresponding business objects. Because co-models are co-ordinated by their corresponding business objects, several intelligent decisions can be made concerning the simulation of the co-model based on the current state of the co-model and the business rules that were stated prior to the simulation

Anatomy of a business object

The building blocks of business objects are listed below and further analysed subsequently.

- *Recorder:* A facility that enables the monitoring of business process operation through recording its performance and transactions.
- *Business Rules*: Strategies and policies of organisations that concentrate on specific business concepts and realised by business objects.
- *System Dynamics Co-model:* System dynamics models that represent specific business concepts and are formed in a system dynamics notation.
- *Methods and attributes:* The actions that the business object can perform, the characteristics that distinguish it from the rest of the business objects and the information that can communicate to the rest of the business objects.
- *Scheduller:* In order to handle simultaneous operations, business objects schedule their activities and execute them in a timely manner.

Each business object contains a facility called a *Recorder,* realised by an object that is aggregated within each business object. The main purpose of the recorder is to maintain activity logging records of the object’s actions (monitors business objects), which provides an audit trail for all corporate processes, information that is important for the stochastic aspects of the simulation. The recorder is mainly active during system operation, where the operational aspects of the business objects are utilised. During that time information concerning the actions of the business object is registered in a time-oriented sequence together with information concerning their duration, service and execution time. As a result the simulation can be supplied with alternative data in order to bring the state of the model closer to the desired state during simulation. According to the required and the current input data supplied to the simulation, the various business objects can identify the required modifications to the simulation parameters in order to satisfy the business policies as captured in the business rules.

Information concerning the activities of each business object are used to create patterns of behaviour. Knowing that certain actions occur at certain times on certain days and have a certain duration, provides important stochastic information in all aspects of the business operation. Simulation models can utilise this type of information for accurate operation.

Business rules represent the intrinsic relationships between business entities. Business rules as a way to specify business logic have the advantage of combining automatic executability with a relatively high level of human understandability, i.e., a high conceptual level and a "declarative" (rather than only procedural) semantics. The latter enables non-programmers, especially business-domain experts such as marketing managers, to specify business rules, and to modify them relatively easily and often (Kilov et al., 1997).

In the proposed methodology business rules are used to specify the constraints that govern both business objects and simulation models. Constraints in business objects enforce the operation of business processes in pre-specified borderlines. In the same manner simulation is influenced by the same business rules. Business rules may sometimes be realised by simulation models as targets to be reached or borders to be satisfied.

The third element of a business object is the *simulation model*. As stated before this model is realised as a system dynamics model and focuses on the same business concept as its encapsulating business object. Since the model is not operation independent it is important to feed it with the necessary data that will enable its enactment. Such data are normally supplied from collaborating business objects. The model is governed by business rules which are transformed into equivalent numerical format that influence variables in the model. Specific parameters of the model are public to other simulation models in order to preserve the feedback loop paradigm of system dynamics.

The *methods and attributes* of business objects are used for the operational aspects of the business process. Some attributes though are used to publish the state of variables in the simulation model to interested business objects. In addition communicating business objects can enforce updates to the state of variables of other business objects simulation models.

Finally, the scheduler object is responsible for the organisation of the business object’s activities in order to fully utilise the object’s capacity. The scheduler schedules the activities planned for execution by its embedding business object. Intelligent decisions can be made concerning the sequence of activities. The scheduler can dynamically enact a simulation process in order to identify the object’s expected workload and accordingly schedule the object’s activities. Since the simulation of the organisation’s business processes would reveal what activities need to take place, each scheduler could use the information supplied by the simulation model during business operation. For instance if the outcome of simulation reveals that a batch of activities will appear in a periodic fashion, the scheduler can schedule the object to become available for the times and duration of these activities.

Business Objects Communication

As stated before, each business object incorporates a corresponding simulation co-model. Simulation can not be completed without the co-ordination of all participating business objects. Necessary simulation parameters of each co-model are passed to and from its encapsulating business object. Consequently these parameters are passed by the business object to the next business object in the loop which in turn passes the parameters to its embedded co-model. At the same time business rules direct the simulation towards the desired state of the system. These rules are translated into simulation notation prior or during simulation and are passed to the system dynamics model. Business rules act as coefficients to various elements of the model and influence the behaviour of the model in a variety of ways. Figure 2 illustrates two business objects and the communication between these two objects in order to satisfy the wholicity property of the system dynamics model. Each co-model concentrates on a specific business aspect which is realised by its encapsulating object. The interrelationship of the two co-models thought remain active by requesting their encapsulating objects to pass or receive certain parameters necessary for the continuity of their operation.

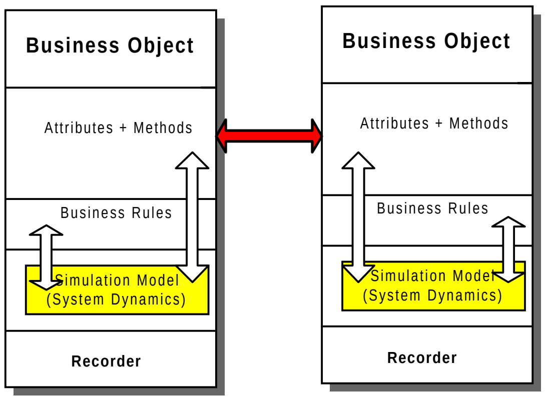
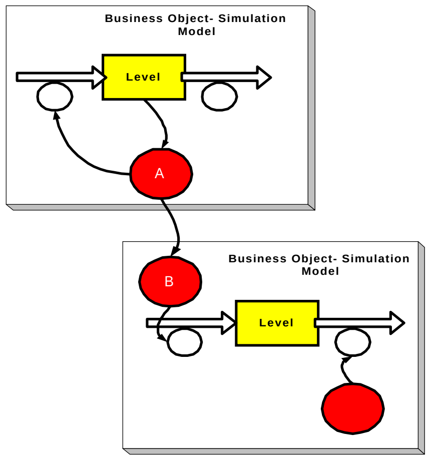


Figure 2 Business Objects communication

Figure 3 illustrates the communication of two co-models. The figure simplifies the communication by directly drawing the connecting arrow between the two co-models. The variable "A" of the first system dynamics co-model influences variable "B" that resides in a different co-model and as a result in a different business object. Each simulation co-model's variable is directly monitored by the encapsulating business object. So in the above case variable "A" would have been communicated by the governing business object to the interested business object which in turn would have pass it through to the variable "B" in its embedded co-model.


Business Object- Simulation Model
Level
A
Business Object- Simulation Model
B
Level
Figure 3 Business object's simulation co-model communicating with a simulation co-model of a co-operating business object

Business Rules

In order to make business rules active participants of the simulations model their transformation into simulation notation is necessary. Blending into the simulation as coefficients, business rules enables a guided simulation towards set goals. This is a form of intelligent decision making during the simulation process. Simulation behaves as an organisation operates in a fast forward notation. Management decision is also functioning in a fast forward manner. Simulation could change direction of operation as a result of an intelligent decision based on the business rules. The intelligent decision is recorded during simulation and presented with a time oriented perspective to the management.

Additionally business rules can influence the structure of the simulation model. This can be achieved in the same manner that transistors operate in a electronic circuit. By preventing or enabling the flow of electric current through certain paths of the circuit, they alter the circuits behaviour. Since system dynamics models are based on the same principles (flows and rates) the same effects could be obtained. Figure 4 illustrates the effect of business rules on the system dynamics model.

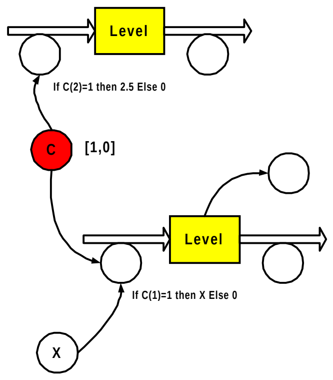


Figure 4 Business rules realisation

The control array "C" represents a business rule. This array can control two flows. Each element of the control array represent one of the two incoming flows in the model. The number "1" in the array enables the flow of data from the incoming flow to the level. Zero on the other hand prevents this action. By preventing the flow of data in each of the two flows business rules deactivate part of the system dynamics model. Designing a model that is adaptable enough enables simulation of dynamic model structures, a characteristic beneficial for a guided simulation process. In the above example the lower flow is activated while the upper flow is deactivated since C(1) is one and C(2) is zero.

In addition to the above approach business rules are also realised in the simulation model variables using the Powersim built in scripting language (Powersim, 1996) Since the variables are directly linked to business concepts through the embedding business object, business policies can be communicated to the model directly. For instance flow variables are weighted accordingly in order to reflect the policy that needs to be reinforced.

Architectural advantages

Integrating two powerful technologies like those used in our approach combines the benefits of both. The advantages of such an integration are analysed in the following sections:

Interface perceptiveness

Integrating business objects with system dynamics provide an intuitive interface to the simulation engine based on real world concepts. Simulating a business process requires taking into consideration a large number of information entities that interact with each other at a frantic rate. On the other hand, the art of simulation is the ability to simulate a model on a variety of scenarios. In order to achieve this it is necessary to translate the business scenarios into simulation notation. During this process a considerable number of simulation variables must be populated, a task that is particularly time consuming. Business objects can assist in this task by providing a natural interface to the user and also reducing the complexity of the simulation infrastructure. This is due to their ability to abstract from business concepts ignoring the intricacies inside them.

Business Oriented modelling

System dynamics models can get quite complicated, despite the fact that they are extremely powerful and assist individual to comprehend the business domain. This makes them not attractive to business people (Lyneis, 1999).

Since high level specification is easier to understand and maintain than low level code, separation from technical details reduce the “surface area” to be investigated. As a result because of their concrete business semantics business object reduce the intricacies that exist in the actual simulation model by presenting to the user only the information that is needed to utilise the model.

Combined Business process and simulation models.

Since business object and system dynamics concentrate on the business process models of the organisation, it is wise to combine both methodologies in order to avoid duplication of business process models.

Since business objects are the main building blocks of a business processes and represent real business concepts, they are ideal tools for the representation of business processes. Integrating business object with simulation objects enable the elimination of duplication of business process models for simulation and enterprise operation. Moreover due to the ability of business objects to accumulate historical information during business processes operation, they enable the simulation of various business scenarios on more accurate stochastic information.

With the use of a *recorder* encapsulated in every business object of the process model, information concerning the performance, resource consumption, task completion time, popular tasks, not popular tasks, peak time, off peak, frequency of task utilisation and so forth can be feed to the simulation model dynamically and provide the simulation objects with additional information required for the probabilistic simulation of the model.

Scalability

The simulation models can be easily extended by attaching additional business objects to the existing structure. Businesses evolve over time, due to changing business requirements. This means that their information systems must also evolve to keep pace with the new requirements. Applications that are based on the business objects paradigm are scalable and easily modifiable. Since the simulation model is embedded in the business objects themselves there will be no extra effort to recreate the new simulation model for the new reorganised business process.

Goal Seeking Simulation

One of the main ingredients of business objects are the business rules which govern the operational and simulation aspects of the process model. Information concerning the organisation’s policies and restrictions are fed into the simulation model, which consequently attempts an adaptive simulation session in order to keep the state of organisation in the policies context and if not to specify sequence of actions to be followed in order to bring the organisation to the desired state. This can be achieved by continually comparing the state of the simulation with the desired state realised in the business rules.

Comparative Simulation

Since the output of simulation is stored in the business objects themselves, various comparisons can be made between business variables that enable visualisation of the various side effects that a business change might have to the organisation. The same result could not had been realised in a straight forward fashion with a system dynamics simulation tool. Moreover the flexibility of presenting the output of simulation in a variety of styles, enables a more informative view of the organisation's state.

Backtrack simulation

Sometimes the best way to investigate the cause of a problem is to go back in time step and view the changes in the system in a backward motion. Since organisations produce their own path it is relatively easy to get all the necessary information in order to backtrack the state of organisations until the point of diversion is reached. Since business objects are ideal for storing information related to the business aspects they represent, they could use such historical data in combination with the simulation model that they encapsulate to recreate past behaviour of organisations. Simulating the organisation in a backward manner would enable the identification of the diversion point in the business processes. This point can give important guidelines for the resolution of the problem in hand.

Implementation Architecture

The above architecture is realised using two development tools, the Powersim constructor (a tool for developing system dynamics models) (Powersim, 1996) and Microsoft Visual Basic 6.0. The former is used to construct the system dynamics model and the latter to develop the business objects infrastructure together with the user interface (Lhotka, 1997). The two technologies were integrated with the use of an ActiveX control provided by the Powersim Engine, a program that allows simulation of models by an application without the need to invoke the Powersim constructor.

An overview of the implementation architecture is illustrated in the figure 5 below.

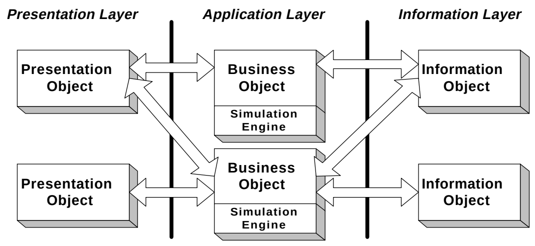


Figure 5 Implementation architecture

The presentation layer of the architecture is realised with the user interface objects which represent the mean of presenting the state of the system to the user and communicate the users requirements to the system. The next level down is the application layer which is composed of the business objects that define the logic of the application. This layer also encompasses the simulation engine of the system. The business processes are realised in this level. The last layer is the information layer which represent the means of storage for the information that resides in the business objects.

Presentation Layer

For the presentation layer Visual Basic 6 is employed with a number of ActiveX controls (Appleman et al., 1998). The structure of this layer is presented in figure 6. The document viewer represents the container for the alternative presentation objects. These are realised in the form of ActiveX documents. Presentation objects are loaded into the viewer which is customised according to the object in hand. Presentation objects are employed with the necessary validation methods that preserve the objects integrity. Business rules encapsulated in the presentation objects navigate the user to the right sequence of events in order to achieve a certain task.

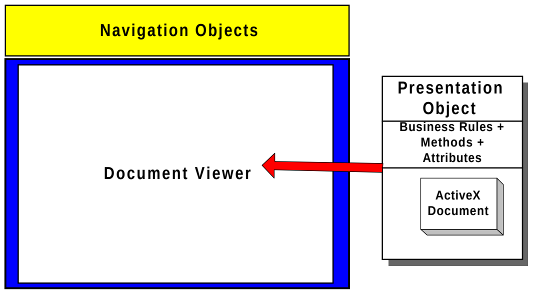


Figure 6 Presentation layer

Application layer

The application layer has been developed using Visual Basic 6. This level contains the business logic of the architecture. Business objects act as servers for the presentation layer. Presentation objects invoke events in the application layer which are realised by the business objects. Business objects are realised in a form of dynamic link libraries (DLLs) (Appleman et al., 1998). The simulation model in each of the business object is preserved in an ActiveX control provided by Powersim Corporation (Powersim, 1996). The control, named Powersim engine, is encapsulated in each business object. Business objects can invoke the properties and methods of the control through its application programming interface (API). Simulation models are assigned to the Powersim engine ActiveX controls prior to runtime and manipulated directly by their governing business objects (figure 7).

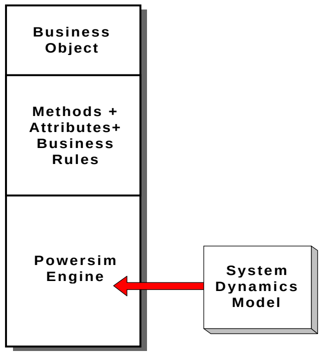


Figure 7 Business object architecture

Simulation is synchronised by a control business object named the co-ordinator which passes messages to every business object in order to invoke their simulation model. The co-ordinator is essentially a business object that sends messages to its governing business objects in a sequential manner in order to give the impression of entirety of the simulation model. The technique is based on the time sharing principle that was employed in computer networking. Each computer is assigned a predefined time slice during which it has full access of a common processor. When its time slice expires the processor’s utilisation is passed to the next consecutive computer in the network. Due to the small time slice each user have the impression that he has complete utilisation of the processor (Silberschatz et al., 1998). Figure 8 illustrates the functionality of the co-ordinator. Messages are send in a round robin fashion to each business object. Each business object simulates its embedded co-model for one time step at a time (the co-originator invokes one object at a time). Each object receives the outcomes of the preceding object (in the time sharing loop), process them in its encapsulating simulation model and communicates the results to the next object (according to the co-ordinator's schedule). The arrows between the business objects represent the communication of the simulation parameters. The arrows between the co-ordinator and the business objects denote the invocation process.

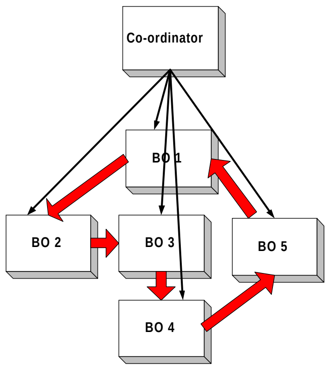


Figure 8 The co-ordinator object is responsible for directing the simulation process

Information layer

The information layer concentrates on the physical storage of data. This layer is realised using the MS Access database. The various tables of the system are interrelated in a normalised way to prevent duplication and preserve integrity of information (Roman et al., 1999). This layer supports the business layer by providing the information for its operation. Historical information collected during operational activities, registered with the recorder, is also located in this level. This data is analysed by specific objects in the application layer to provide statistical information important for simulation, business process improvement, or even business process reengineering.

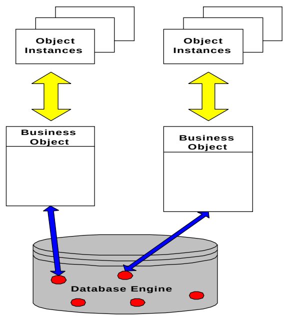


Figure 9 Direct mapping of database record to business object's instances

According to figure 9 information in the database engine are mapped directly to business objects. Records of this information are represented by object instances in the application layer.

Conclusions

System dynamics is one of the most mature simulation techniques based on the principles of system thinking. System dynamics simulation employs computer techniques to extract salient features of complex business processes. Business Objects on the other hand are building blocks of business processes. Systems Dynamic simulation and business objects have therefore similarities as both are concerned with business processes, the former by simulating them and the latter by operating them. Combining system dynamics and Business Objects seems therefore an ideal way of integrating simulation and operation activities in an organisation. The simulation of the organisation’s business processes is based on business rules that are encapsulated in business objects.

References

Appleman, D., Appleman. D. (1998) Developing Com/Activex Components With Visual Basic 6, SAMS .

Cover, J. (1996). Introduction to System Dynamics, Powersim Press, Norway

Kilov, H., Simmonds I. (1997) Business rules: from business specification to design, ECOOP'97 workshop on Precise Semantic of Object Oriented Modelling Techniques.

Lhotka, R. (1997) Professional Visual Basic 5.0 - Business Objects, Wrox Press , Canada.

Lyneis, J. (1999) System Dynamics for Business strategy: a phased approach, System Dynamics Review , Vol. **15** , 1 , 37-70.

Powersim (1996) Powersim 2.5- Reference Manual, Powersim Press, Norway

Roman, S., Petrusha R. (1999) Access Database Design & Programming, O'Reilly & Associates , Ed. 2 .

Silberschatz, A., Galvin P. B (1998) Operating System Concepts, John Wiley & Sons , Ed. 5