Mathematical Modelling and Analysis
Volume 5, 2000, pages 7–17

THE SIMULATION OF DYNAMIC SYSTEMS USING COMBINED MODELLING

R. ALZBUTAS, V. JANILIONIS

Kaunas University of Technology
Studentų str. 50, LT-3031 Kaunas, Lithuania
E-mail: robertas@isag.lei.lt, vyjan@fmf.ktu.lt

Received September 30, 1999

ABSTRACT

The new approach to the problems of dynamic systems simulation is proposed. The analytical and imitation modelling of non-linear complex dynamic systems which comprise simulation of continuous and discrete processes with constant and variable parameters, are investigated. The aggregate mathematical modelling scheme [1] and the method of control sequences for discrete systems specification and simulation are used as well as the dynamic mathematical modelling scheme for continuous process formalization and modelling. According to them the investigated systems are presented as the set of interacting piecewise linear aggregates, which can include processes described with differential equations. The above mentioned approach is used in developing software for the construction and research of the models. The modelling of the dynamic systems' control is also analyzed and developed software for the dynamic systems' simulation is presented. It is related to the proposed combined modelling methodology. The developed dynamical simulation system ADPRO (Automatic Differentiation PROgram) extends applicability of the system SIMAS (SIMulation of the Aggregate Systems) [2] with dynamical simulation means realized with APL2 (A Programming Language 2) and based on automatic differentiation [3]. The created model of service process and its control can be used as a base for other models of wide class complex dynamics' systems [4], the parts of which are described with differential equations.

1. INTRODUCTION

In order to use and control various dynamic processes, more and more complex control systems are being created. Simulation becomes a main part of hybrid dynamic system (HDS) analysis. Often it is the only possible way to investigate continuous and discrete system control. There is a need for special simulation systems in which the models of HDS components can be created, interconnected and changed as the system evolves. Such components can have hybrid behavior, consisting of continuous-time phases separated by discrete-event transitions. Components can be developed and evolved independently, or they may interact through their inputs and outputs according external and internal events. The low and conditions of dynamic process can be changed in simulation time.

The simulation can be used for: analysis of system states in different control conditions; analysis of discrete events statistic and system states dynamic (in order to estimate reliability, optimize various processes and their control, etc.) The main problems of HDS simulation are the disadvantages of aggregate mathematical modelling software and its application, the modelling problems of non-linear complex HDS and the disadvantages of simulation of system's control an observe process.

In order to solve the problems of complex system model construction and realization we use aggregate mathematical modelling scheme (A-scheme) based on the set of interacting piecewise linear aggregates (PLA) and a method of control sequences. The above approach is used in developing software for the construction of models. The simulation system SIMAS (SIMulation of the Aggregate Systems) has been developed at Kaunas University of Technology and represents the software of this type. The experience of usage of aggregate mathematical modelling scheme in the development of system models, which separate parts are described by differential equations or their systems, has demonstrated the disadvantage of this scheme.

In other case, the dynamic mathematical modelling scheme (D-scheme) based on differential equations and their systems is more useful for continuous deterministic system modelling. So it needs the tools which could ensure the solution of general m-th order differential equations and general systems of first-order differential equations. For this purpose there is used and developed automatic differentiation method [3; 5], which provides an elegant and conceptually simple solution of an ordinary non-linear different equation. Automatic differentiation is an untraditional approach to differentiation. While symbolic differentiation uses "the rules" with symbols, automatic differentiation uses "the rules" with values. Calculations are performed in a system of vectors.

The ability of APL2 to "think" in terms of vector operators makes it an ideal language for implementation of automatic differentiation. In contrast to other languages, the APL2 code of automatic differentiation is very simple. The large and expandable library of operators and functions and the language orientation to the operation with vectors imitate mathematical symbolism in APL2. These mentioned features fit in automatic differentiation very well [6].

2. SIMULATION OF THE AGGREGATE SYSTEMS

In the application of the aggregate approach for system specification, the system is presented as a set of interacting PLA. For this part of investigations the theoretical background is PLA formalism, which permits to use the single formal specification for the development of the models. It is very important for design of distributed and complex systems [1; 2].

In practice one of the main steps of complex system aggregate model creation is defining the model structure. An example is presented in fig. 1. A lot imitating complex systems consist of separate autonomous parts, which exchange information without any loss and distort. In such case each part (aggregate in A-scheme) is created separately and independently from mathematical model of other parts. So any aggregate (as example Service Unit in fig. 1) can be realized using D-scheme, i.e. using differential equations.

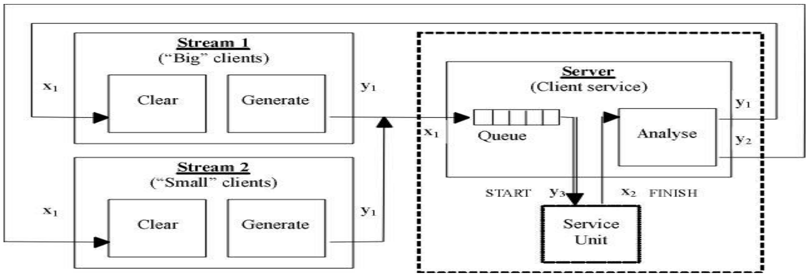

Figure 1. An example of system aggregate model structure.

In fact the A-scheme and the method of control sequences for discrete systems specification and simulation at the same time are used with the dynamic mathematical modelling scheme for continuous process formalization and modelling. The investigated systems are presented as the set of interacting PLA, which include processes described with differential equations. The above mentioned approach is used in model specification and implementation with developed software. Formal specification of each aggregate is used for the construction and research of simulation programs. Below there is an example of aggregate specification, which presents description of one aggregate from analyzed service system with hybrid process of service unit [5; 6].

Aggregate: Server
1. Input signal set: $X = \{x_1, x_2\}$;
$x_1 =$ (Client), - signal Client (request for service);
x_2, - signal FINISH (end of service in Service Unit);
2. Output signal set: $Y = \{y_1, y_2, y_3\}$;
$y_1 =$ (Client), - signal Client (end of service of client);
$y_2 =$ (Client), - signal Client (end of service of client);
y_3, - signal START;
3. External events set$E' = \{e'_1, e'_2\}$;

e_1', - incoming of signal x_1;
e_2', - incoming of signal x_2;

4. Internal events set: $E'' = \{e_1'', e_2''\}$;
e_1'', - finish of transmission of Client to Service Unit;
e_2'', - finish of service and analyze of Client;

5. Control sequences: $e_1'' \to \{\xi_i^1\}, i = \overline{1,\infty}$; $e_2'' \to \{\xi_i^2\}, i = \overline{1,\infty}$;
ξ_i^1, - random duration of Clients transmission,
ξ_i^2, - random duration of analyze of Client,

6. Aggregate state:
Continuous part: $W(t_m) = \{w(e_1'', t_m), w(e_2'', t_m)\}$;
$w(e_1'', t_m)$, - moment of transmission end;
$w(e_2'', t_m)$, - moment of service end;
Discreet part: $V(t_m) = \{Q(t_m)$, State$(t_m)\}$;
$Q(t_m)$, - length of Client queue;
State(t_m), - information about Client;

7. Initial state:
$w(e_1'', t_0) := \infty, w(e_2'', t_0) := \infty$, - undefined moment of events,
$[Q(t_0)] := 0$, - queue length at initial time;
State$(t_0) := nil$, - state of information absence;

8. Transition and output operators (for each external event):

```
$H(e_1')$: if State($t_m$) $= nil$                       - state of information absence;
        then $w(e_1'', t_{m+1}) := w(e_1'', t_m) + \xi_i^1$;
             State($t_{m+1}$) :=(Client);
        else $w(e_1'', t_0) := \infty$                    - undefined time moment;
             _Q_PUT($Q(t_m)$,(Client)),                - put client in to queue;
$G(e_1')$: $Y := \varnothing$,                                  - empty set of output signals;
$H(e_2')$: $w(e_2'', t_{m+1}) := w(e_2'', t_m) + \xi_i^2$,       - activate internal event;
$G(e_2')$: $Y := \varnothing$,                                  - empty set of output signals;
$H(e_1'')$: $w(e_1'', t_{m+1}) := \infty$,                     - undefined time moment;
$G(e_1'')$: $y_3$ :=PARAM(Client),                       - send Client request;
$H(e_2'')$: if $[Q(t_m)] > 0$                              - if queue is not empty;
        then _Q_GET($Q(t_m)$,(Client)),                - get Client from queue;
             State($t_{m+1}$) :=(Client);
             $w(e_1'', t_{m+1}) := w(e_1, t_m) + \xi_i^1$;
        else State($t_{m+1}$) $:= nil$;                  - state of information absence;
$G(e_2'')$: if SR(Client)=1,                             - if Client is from 1st source;
        then $y_1$ :=(Client),                          - return Client to $1^{st}$ source;
        else $y_2$ :=(Client),                          - return Client to $2^{nd}$ source.
```

SIMAS (see fig. 2) is the simulation system, which is using aggregate specifications. This system is used for the automatic development of the simulation models and their analysis. SIMAS has been developed on the basis of aggregate mathematical modelling scheme. The simulated system is divided into subsystems. The subsystems are described in accordance with the mathematical aggregate scheme, and their interconnections are described in accordance with the aggregate connection rules. Using those descriptions SIMAS creates the programs of simulation models automatically.

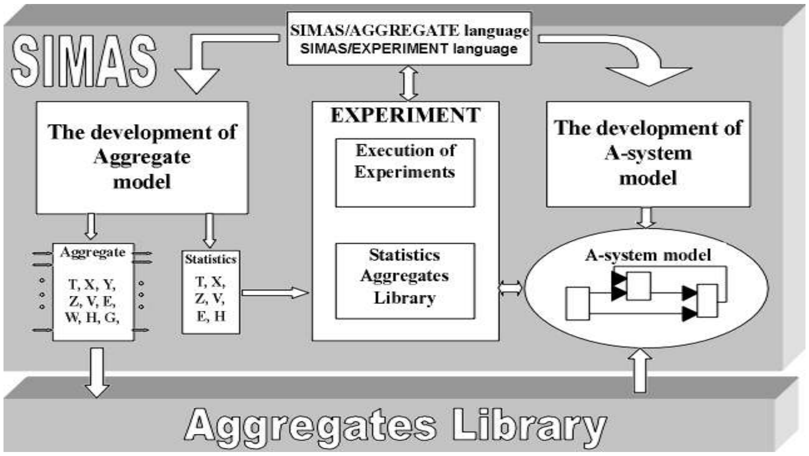
SIMAS
SIMAS/AGGREGATE language
SIMAS/EXPERIMENT language
The development of Aggregate model
EXPERIMENT
Execution of Experiments
Statistics Aggregates Library
The development of A-system model
Aggregate
T, X, Y, Z, V, E, W, H, G,
Statistics
T, X, Z, V, E, H
A-system model
Aggregates Library

Figure 2. The structure of simulation system SIMAS.

SIMAS consists of three subsystems:

- Subsystem of the development of the aggregate models is a program with different tools, which enable to develop programs of aggregate model.
- Subsystem of the development of A-systems model inputs and uses description of A-system. An output of this subsystem is a simulation model, which is formed from the elements of aggregate library.
- Subsystem of execution of experiment executes the tasks using the program written in SIMAS EXPERIMENT language. The subsystem sequentially performs all the aggregated system investigation stages: simulation, data collection, analysis and reporting.

3. SIMULATION OF DYNAMIC PROCESSES

As usually for modelling dynamic continuous process there are used n-order differential equations. The solution of differential equations can be various functions. In order to use the general function form it is convenient to get it as Taylor row. In such case the main problem is to calculate the high order derivatives $y^{(k)}(x)$, then $k >> n$. For solving this problem we use the automatic differentiation method [3; 4].

Numerical methods could be used to obtain approximate answer rather quickly, but the difficulty of obtaining reasonable accuracy grows with the order of the derivative. Automatic differentiation is a quite different approach that is theoretically exact and easy to use in practical computing. Automatic differentiation makes it possible to evaluate high-order derivatives of function at a point with accuracy equal to that provided by symbolic differentiation, but at a fraction of the cost in computer time and memory. So from a practical viewpoint, the method is unmatched in its ability to calculate the value of high-order derivatives at a point.

Automatic differentiation is interesting in mathematical point of view and, when it is implemented in APL, it is a powerful, accurate, clear, and practical tool for finding the numerical value of derivatives at a point. The main calculations are performed with vectors in following form $(y, y'_x, y''_x, ..., y^{(k)}_x)$. This vector is counted by combining the formal rules of differentiation with numerical evaluations of derivatives as they are produced step by step. The row elements are being recursively generated and Taylor row is found from them. So for various implementations of automatic differentiation it is also possible to manipulate with one common expression - the Taylor row coefficients or the derivatives vector of analyzed function $(y, y'_x, y''_x, ..., y^{(k)}_x)$.

All above presented features are very useful in implementation and simulation of dynamic process. It is necessary to mention that the using of automatic differentiation does not give special possibilities to analyze the convergence of counted power row, because in most cases it is impossible to get analytical expression of row n-th element. This method is very proper if exists solution of the differential equation expressed in power row. Automatic differentiation can be used very successfully in some calculations of engineer practice and various experimental researches where it is possible to estimate the solution empirically.

The method of recursive evaluation of higher-order derivatives by using a vector operation is the base for created dynamical simulation tools. The main functions of dynamic simulation system ADPRO are to calculate high-order derivatives of function at a point, solve general m-th order differential equations and general systems of first-order differential equations and generate Taylor polynomial coefficients. All these calculations can be used in other software. For ADPRO interaction with user and other simulation systems it is implemented various data exchange means and created special interface. According to the performing functions and distribution of information the structure of simulation system ADPRO can be divided into five subsystems and three libraries (fig. 3).

The external task library consists of separate task files. The internal task library is used in order to save the tasks formulated by dialog between system and user. The external result library saves the main results, which can be used as data in other systems. In order to exchange with information among different libraries, subsystems and users of the system, there was developed a language DIFLAN. The differential equations used in ADPRO system are described according to the operators and special rules of this language. The form of m-order differential equation is:

$$(DYm) = F(X, Y, (DY1), (DY2), ..., (DYm-1)),$$

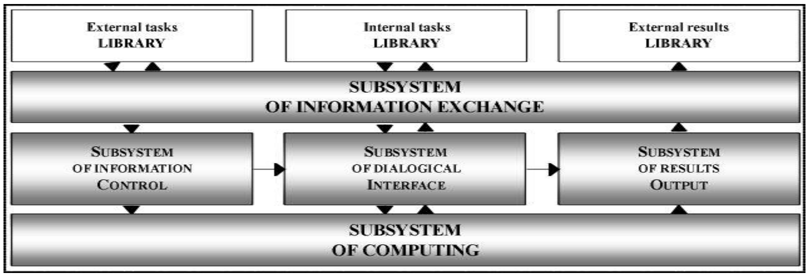

Figure 3. The dynamic simulation system ADPRO (developed using APL2).

where (DYk) denotes k-th derivative of Y according X. In order to express the function F there are used variables: $X, Y, (DY1), ..., (DYm - 1)$; special operators and rules of DIFLAN. For m system of differential equations is used the following form:

$$\text{dY[1;]/dX} = \text{F1 (X, Y[1;], ... , Y[m;])},$$
$$...$$
$$\text{dY[m;]/dX} = \text{Fm (X, Y[1;], ... , Y[m;])},$$

where $Y[k;]$ corresponds to k-th function system $Yk(X)$. In order to express function Fm there are used the same operators and laws as in expression of differential equation, but in this case there are used new variables: $X, Y[1;], ..., Y[m;]$. Such form is the same as one dimension APL2 matrix, as the base of DIFLAN rules is APL2 language. In contrast to other languages, the APL2 code of differential equation solving using automatic differentiation is not so complicated [6]. A large and expandable library of operators and functions and the language orientation to the operation with vectors fit for dynamic process simulation as well as for interface needs.

As a rule analytical and imitation modelling is used separately. A preliminary decomposition of real system to subsystems and the usage of the most suitably modelling method enables to avoid big generalization or pettiness of the model. Combined (analytical & imitative) modelling joins both methods advantages and avoids many of their disadvantages [5].

In this case for modelling there are needs for specific mathematics methods and software. At first created aggregate simulation system SIMAS was joined with another simulation system ADPRO (fig. 4). For this purpose at first their was developed the system of automatic differentiation [4] which was expanded to the universal dynamic simulation system ADPRO. This system can be managed with dialog or with data files. The results and initial data can be presented directly on the screen in case of dialog regime, or in data and results file, in case of autonomic work regime. The main purpose of autonomic work regime is the integration of system ADPRO to other simulation systems. Subsystem EXEC and AD use this regime.

The integration subsystems must ensure the links between the system ADPRO and other programs in PASCAL language (as example the system SIMAS).

The subsystem AD forms the task for the ADPRO according to the received requirements; i.e. AD forms the external task library. Then received results from system ADPRO are written to the external results library. Then if need subsystem AD solves another problems and gives the solutions to initialization program SIMAS.

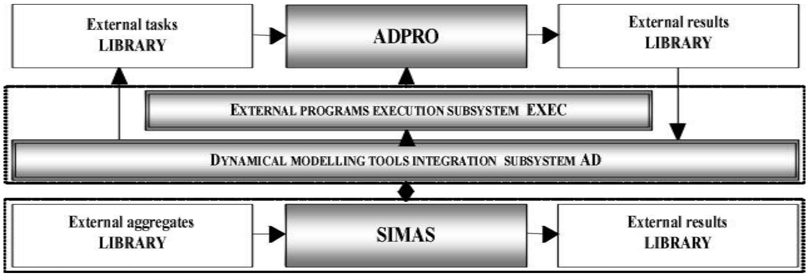
External tasks LIBRARY
ADPRO
External results LIBRARY
EXTERNAL PROGRAMS EXECUTION SUBSYSTEM EXEC
DYNAMICAL MODELLING TOOLS INTEGRATION SUBSYSTEM AD
External aggregates LIBRARY
SIMAS
External results LIBRARY

Figure 4. The process of model design and modelling phases.

4. CONTROL AND OBSERVE SIMULATION

The general control system can be a part of dynamic systems in which control is expressed in continues laws - differential equations. The model of process control often is divided into two separate parts: control subsystem and dynamic process. Currently investigations are performed during specification and simulation of complex control systems with continuous and discrete control and process components. When the general control scheme unfit, the dynamic processes are observed and managed according to the logical laws. An example, can be a service unit in which the process is stopped in every observation and if the process reaches certain conditions, the manager can change the law of process functioning [6].

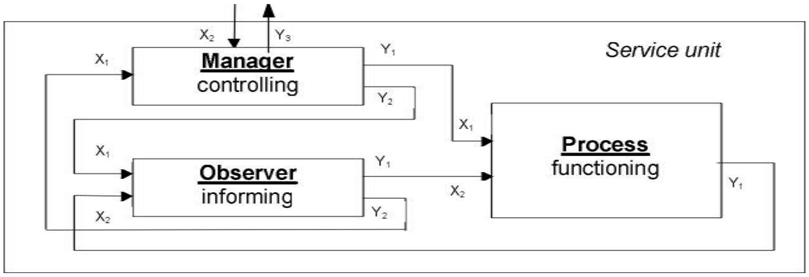

Figure 5. The scheme of aggregate model for general control system.

The created service unit (fig. 5) is independent aggregate system, which can be an extension of other systems, e.g. the extension of aggregate system presented in fig. 1. The structural scheme of whole aggregate model consists of such aggregates: Stream1, Stream2, Server, Manager, Observer and Process. All aggregate relations and corresponding signals are specified according to the given method. A conceptual model and the informal description of each aggregate must be presented in terms of aggregate approach.

For example it is possible to model a service unit, which at first works according to one law, and after it reaches the defined state - according to second law. Each law of continuous process with changeable parameters can be defined as differential equation or their systems. The investigated process in the first part where $t \in (T_0, T_1)$ is described by differential equation, which depends on the time t and changeable parameter K. The parameter K represents the intensity of service and itself can be dependable on service request Y (as example $K = 3\sqrt{Y}$).

In order to show the possibilities of combined modelling approach as the first part of process we have to use equation $\frac{dy}{dt} = K \cdot t$, which common solution is $y = \frac{K}{2} \cdot t^2 + C$. Lets analyze the case when $C = 0$, i.e. it is valid condition: $y = 0$, when start time moment $t = 0$.

The service law must be changed at the moment Tx, when service process has been reaches the state $y = Y/2$. As example of the law of the second service part could be equation $\frac{dy}{dt} = K \cdot (T_x - t)$, which common solution is $y = K(T_x \cdot t - \frac{t^2}{2}) + C$. If again $C = 0$ and imitation start time moment $t = 0$, then service process would continue at the same state $y(T_x)$.

In the investigated system at the time moments $T \in \{T_0, T_x\}$ the manager change the law of y velocity, i.e. the y'. The management can delay the random time interval, which is distributed according to the uniform law in some interval, as example [0, 2K/60]. In the service process delay interval, its state leaves the same (fig. 6 left part). The modeled service system with a lot different delays in each aggregate can imitate the delay of observing, which also influences service process prolongation. It is possible to simulate observing intensity, which dependents on current service process state. In this case the observing probability could be $p(t) = y(t)/Y$. Then observing intensity is as high, as process state is close to the some value Y and at the same time the process delay is as long, as high is observing intensity.

All processes in service system can be executed as long as we continue the experiment of the system imitation. The service unit functioning dependents on the intensity of client's streams and server work. So in order to investigate whole service system functioning is necessary to simulate the whole system. According to the aggregate approach there are possibilities for the appropriate aggregates interconnections and analyzes of the whole modeled service states $z(t) \in Z$, where $t \in [0, T_1]$. An example of states graphic, when $T_1 = 120$ is presented in fig. 6. Time T_1 is process finish time ($T_1 \in T$).

According to the last graph (fig. 6 right part) we can conclude: in the model the service velocity y' (slope of curve) dependents on client request Y; the process control and observe delay corresponds to provided features. In this case the delays between different client services dependents on inten-

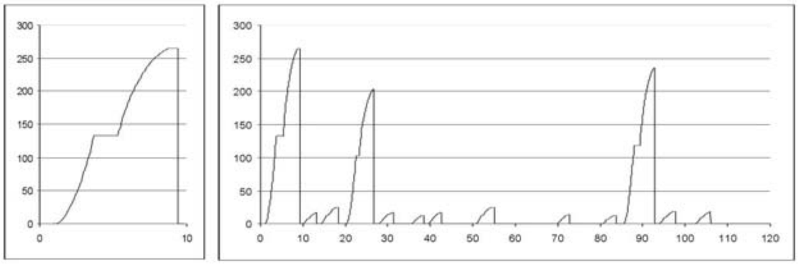

Figure 6. The graphics of modeled service process states in different interval size.

sity of service streams, server queue states and also client requests. So the decomposition of service system structure and functions enables to simulate hybrid dynamic system with detail analysis of its elements and their interconnections. It is possible to conclude that combination of mentioned methods provides an elegant and conceptually simple solution of dynamical systems and their control simulation problems.

5. CONCLUSIONS

Developed combined modelling method (based on imitative and analytical modelling methods) joins aggregate and dynamic modelling schemes and extends the possibilities of their usage.

Created simulation software SIMAS+ADPRO using PASCAL and APL2 languages extends modelling of systems with new specific features, which enable to model complex systems with non-linear continuous and discrete processes and their control more simply.

Created service system control model can be used for creation of other models of various complex dynamic systems [4; 6], the parts of which are described with differential equations.

REFERENCES

[1] H. Pranevichus. Formal Specification and Analysis of Distributed Systems. In: *Lecturer Notes in Applications of AI to Production Engineering. Nordic Baltic Summer School'97*, 1997, 269 – 322.

[2] H. Pranevichus and V. Janilionis. Aggregate simulation system (SIMAS). In: *Perspectives of Computer Systems Development, RPI, Riga*, 1985, 147 – 149, (in Russian).

[3] R.D. Neidinger. Automatic Differentiation and APL. *College Mathematics Journal*, **20** (3), 1989, 238 – 251.

[4] R. Alzbutas and V. Janilionis. Dynamic systems simulation. In: *Mathematics and Mathematical Modelling, 2 book*, KTU, Kaunas, Lithuania, 1998, 82 – 87.(in Lithuanian)

[5] R. Alzbutas and V. Janilionis. The extension of Aggregate simulation system SIMAS with means of dynamical simulation. In: *Mathematics and Mathematical Modelling*, KTU, Kaunas, Lithuania, 1999, 71 – 77.

[6] R. Alzbutas and V. Janilionis. Dynamic systems simulation using APL2. In: *Proceedings of the APL'99 International Conference*, Scranton, USA, 1999, 20 – 25.

DINAMINIŲ SISTEMŲ KOMBINUOTAS MODELIAVIMAS

R. ALZBUTAS, V. JANILIONIS

Darbe nagrinėjamas naujas dinaminių sistemų modeliavimo metodas. Ištirti analitiniai ir imitaciniai sudėtingų dinaminių sistemų modeliavimo būdai, tirti tolydieji ir diskretieji procesai. Pateikta nauja dinaminio matematinio modeliavimo schema, skirta tolydžių procesų formalizavimui ir modeliavimui. Sukurta programinė įranga, leidžianti sudaryti ir tirti tokio tipo matematinius modelius. Taip pat tirtas dinaminių sistemų valdymo modeliavimas bei sukurta atitinkama programinė įranga.