GOAL-BASED INTELLIGENT AGENTS

Zhiqi Shen, Robert Gay and Xuehong Tao

ICIS, School of EEE, Nanyang Technological University, Singapore 639798

zq_shen@yahoo.com, {eklgay, exhtao}@ntu.edu.sg

ABSTRACT

Agents are distinguished for their autonomous and goal oriented characters. To model the complex goals of agents, this paper proposes an agent goal model, namely, composite state goal model, based on Petri Net theory and Object-Oriented methodology. The proposed model not only enables the agent to manage the composite goals, sub goals but also provides the quantitative measurement of partial goals completion. A goal-based intelligent agent model is also proposed in this paper for designing agents based on the goal model. With the proposed goal model and the goal-based agent model, agents are able to present not only behavior autonomy but also goal autonomy. A goal based intelligent business forecasting agent is developed to illustrate the practice of the proposed goal-based modeling and agent design method.

1. INTRODUCTION

Agent technology represents an exciting new means of analyzing, designing and building complex software systems [1]. An agent works towards its goals. The goal-based modeling is one of most important aspects in a successful agent development. This paper explores a new agent goal model, namely, the composite state goal model for modeling the complex goal of intelligent agent. A scenario case in agent oriented business forecasting will be used to illustrate the goal-based modeling throughout the paper.

Business forecasting has been with us since there have been businesses. It plays an important role for decision making in many aspects of businesses. There has been an increasing demand for building business forecasting software systems to assist human beings in managing the forecasting processes [2, 3]. However, the uncertain and complex nature makes it a challenging work to provide software solutions for business forecasting.

Business forecasting includes a set of processes such as data collection, data preparation, forecasting model training and generating forecasting result etc. Nevertheless, most of the current efforts on development of business forecasting software systems are mainly focused on the implementation of specific forecasting methods, i.e. the learning and reasoning algorithms for the specific business forecasting models [3]. They are monolithic systems. They work as individual tools managed by users step by step, going through a single business forecasting process such as training and forecasting. Meanwhile, another group of people are doing research on system implementation,

system integration, process automation, and data collection, etc. Currently with more and more data is published on the Internet and e-services start to play an important role, data collection and system efficiency become more and more important. Data is also a key factor to forecasting accuracy. Obviously existing monolithic systems have limitations to follow the important forecasting principles for managing the whole life cycle of business forecasting and to satisfy the new requirements and new challenges raised by the information exposure over the Internet. Agent-oriented business forecasting systems are emerging as a new approach to designing open, integrated and active business forecasting systems to manage various business forecasting processes [4].

The autonomous, goal-oriented and intelligent characters of an agent make the agent based system a very promising software solution for managing the business forecasting processes. Agent-oriented business forecasting and agent framework for business forecasting agents are discussed in [4][7]. In that work, the goals of agents were modeled using existing task oriented goal model. The business forecasting agents were implemented under many presumptions because it is hard to this kind of goal models to model the agents' goals for real business forecasting applications. In this paper, we present a new agent goal model, the composite state goal model which is based on Petri Net theory and Object-Oriented methodology, for modeling the goals of business forecasting agents. Unlike existing agent goal models, the proposed goal model not only enables the agent to manage the sub-goals but also provides the quantitative measurement of partial goal completion.

Following this introduction, section 2 addresses the related work. Section 3 describes the composite agent goal model. Based on this goal model, an agent model is given in section 4. Section 5 illustrates a goal-based modeling for an intelligent business forecasting agent based on the proposed agent goal model. Section 6 discusses the experimental results. Finally the conclusion is reached in section 7.

2. RELATED WORK

There have been mainly two types of the agent goal models, task oriented model and state oriented model [5].

Task oriented model assumes that agents lives in a task oriented domain, the goal of an agent is a set of tasks to perform. For instance, a notification agent has a goal to notify the customers the new products. The agent's goal is specified as a set of tasks: 1) check if any new product arrives, 2) for each new product, find the customers who might be interested in the new product from customer profile database, 3) notify the customers by sending an email with the new product information to every corresponding customer. Hence the task oriented goal is a fixed list of tasks; the goal is reached only when the agent finishes all the tasks, regardless of the state changes caused by the tasks. Then it will iterate the process.

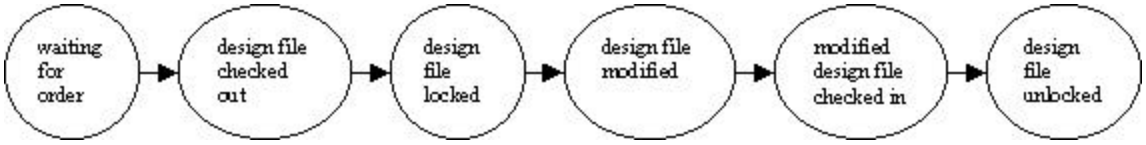

Figure 1 An example of state oriented agent goal model

State oriented model assumes agent lives in the state oriented domain. The agent's environment is evolved with a sequential finite set of the states. A goal of the agent is a final state that the agent tries to reach from its current state by going through a sequence of states. For example, the goal of a manufacturing design agent is to reach the final state by going through a sequence of states specified by a workflow for modifying a design file. The initial state of the design agent is *waiting for an order*, after receiving an order, the agent will *check out* the design file, *lock* the design file, *modify* the file, *check in* the file after modification, and *unlock* the design file, to reach the states showing in Figure 1 respectively. After the goal is reached, the agent returns to the initial state.

As shown, both the task oriented goal and state oriented goal are modeled by a list of tasks and states respectively. The sequence of the tasks or states is fixed, and the agent's goal can only be measured by whether the goal is reached or not reached. Due to the uncertain and complex nature of the real world problems such as business forecasting processes, it is hard to represent the complex goals of an agent simply by a set of tasks list or a set of sequential states. Therefore, a new agent goal model is highly needed.

3. THE COMPOSITE STATE GOAL MODEL

To overcome the limitations of the existing goal models, in this section, we propose a *composite state goal model* for modeling the complex goals of intelligent agents.

The composite state goal model is proposed based on the Petri Net theory [6] and OO methodology. *States* in a composite state goal model are not necessarily modeled in a sequential order. They can reflect state changes in a real application. *Tasks* in the composite state goal model are used to move the agent from one set of states to the next set of states based on the goal model. , Like Petri net, the model is composed of four basic objects: *states*, *transitions*, *arcs* and *tokens*.

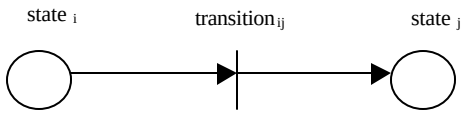

Figure 2 Basic objects of the composite state goal model

As it is illustrated in Figure 2, the object *states*, represented by circles, are used to represent different states that agents need to go through to reach their goals. A state is connected to other states via *transitions*, represented by vertical bars. A transition specifies the relationships between the state objects it joins. Each transition has at least an *input state* and an *output state.* Each transition is associated with a task list that defines the tasks an agent may perform in order to fire a transition. When certain conditions are satisfied, the transition fires, and the agent will evolve from the input states to the output states. The object *arcs*, represented by arrows, are used to connect states to transitions and transitions to states. An arc indicates the relationships between the state and the transition it connects. The object *tokens* are used to present dynamic behaviors of the goal model. When a token arrives in a state, it indicates the state change on that state and the progress of the goal pursuing.

The model can be formally defined as follows:

[**Definition 3.1**] The composite state goal model, Goal-net, is defined as a tuple: CSGM = {$\mathcal{S}$, *S, T, A, C, G, E, L, F, D, N, B, H, R, I, R*$_0$} where:

$\mathcal{S}$ is a finite set of non-empty *types*.
S is a finite set of *states*.
T is a finite set of *transitions*.
$A \subseteq S \times T \cup T \times S$ is a finite set of *arcs* that connect states and transitions.
$C: S \rightarrow \mathcal{S}$ is a node function for each state.
G is a guard function for each transition.
E is an arc function for each arc.
L is a finite set of *task lists.*
$F: T \rightarrow L$ is a task function for each transition.
$D \subseteq S$ is a set of *composite states*.
N is a finite set of *Goal-nets.*
$B: D \rightarrow N$ is a net function for each composite state.
H is level number of the hierarchical structure.
R is a set of time values, also called *time stamps*.
I is an initialization function for each state.
$R_0 \in R$ is the start time.

$\mathcal{S}$ includes types that will be used in the Goal-net. It determines the types, operations and functions that can be used in the net inscriptions (i.e., arc, guard, initialization functions, etc). The node function *C* maps each state, *s*, to a set *C(s)* in $\mathcal{S}$. The guard function *G* maps each transition, *t*, to an expression of type Boolean, i.e., a predicate. Moreover, all variables in *G(t)* must have types that belong to $\mathcal{S}$. The arc function *E* maps each arc, *a*, into an expression which must be of type *C(°a)* or *C(a°)* where *°a* denotes the state that is connected to a transition by the arc *a*; *a°* denotes the state that is connected from a transition by the arc *a*. *L* contains all the task lists that are needed for each transition. The task function *F* maps each task list to a transition on which the tasks need to be executed when the transition is enabled. *N* contains Goal-nets that are sub nets of each composite state. The function *B* maps each composite state of *D* to a Goal-net of *N*. The initialization function *I* maps each state, *s,* into an initial expression which must be of type *C(s).*

[**Definition 3.2**] A state is a pair (*os, r*) where *os* is the state object and $r \in R$ is a time stamp. The state object *os* defines the variables of type *C(s)* and behaviors on the state.

[**Definition 3.3**] A transition is a pair (*ot, r*) where *ot* is the transition object and $r \in R$ is a time stamp. The transition object *ot* defines the properties and behaviors on the transition.

[**Definition 3.4**] A token is a pair (*s, c*) where $s \in S$ and $c \in C(s)$. It indicates the status of a state. When a state holds a token, the state becomes active, otherwise the state is inactive.

[**Definition 3.5**] A composite state is a state, which can be decomposed to a Goal-net where the net starts with a start state and ends by an end state. The following rules apply:

- There is only one start state in a Goal-net and it is initialized to hold a token. There is only one end state in a Goal-net and it is initialized not to hold token.

- The transition $t \in s^{\circ}$ of the start state s is enabled iff both the composite state and the start state start to hold tokens. The notation s° represents the set of transitions each element of which is connected from state s.
- If the transition $t \in s^{\circ}$ of the composite state s is enabled, both the end state and the composite state must hold tokens.
- The net will be reset after any transition $t \in s^{\circ}$ of the composite state s fires.

[**Definition 3.6**] The goal of a composite state goal model is the composite state, which is the root state of the hierarchical structure.

[**Definition 3.7**] A sub goal of a composite state goal model is the composite state, which is not the root state of the hierarchical structure.

There are two kinds of state objects in the composite state goal model: an *atomic state object*, represented by blank circle, accommodates a single state which could not be split any more; a *composite state object*, represented by shadowed circle, may be split into other state objects (either composite or atomic) connected via transitions. Figure 3 presents a hierarchical structure of the composite state goal model. The root composite state object in the highest level of the hierarchical structure represents the overall goal of the agent and the composite state object in lower levels of the hierarchical structure represents the sub goals of the agent. A higher level of composite state objects (goal or sub goals) can be split into lower-level state objects connected via transitions.

There are three types of transition objects in the composite state goal model: *direct_to transition object*, *concurrent_with transition object*, and *jump_to transition object*.

- ***Direct_to***: this type of transition objects designates a direct connection in sequence from one state to other states. It defines a successive relationship between states. For example, in Figure 3, State *i* is connected to State *i*+1 via a direct_to transition. This implies that State *i*+1 should be reached after State *i* expires, in another word, State *i*+1 is a continuous state of State *i*.

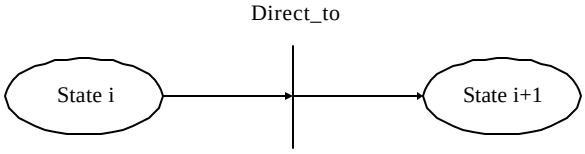

Figure 3 A *Direct_to* relationship

- ***Concurrent_with***: this type of transition objects specifies a concurrent occurrence between one state and another state. It defines concurrent relationship between states. For example, in Figure 4, State *i*+2 and state *i*+2' are two concurrent state objects.

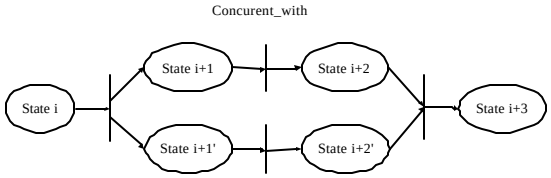

Figure 4 A Concurrent_with relationship

- ***Jump_to***: this type of transition objects specifies a jump connection from one state to other states. It defines a jump relationship between states. In Figure 5, State *i* is connected to State *j* via a jump_to transition, while State *i* is connected to state *i*+1 via a direct_to transition. This indicates agent will jump from State *i* to State j instead of progressing from State *i* to State *i*+1 sequentially. The inhibitor *C1* and *C2* coordinate the behaviors.

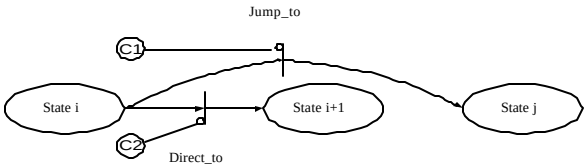

Figure 5 A Jump_to transition

For example, in Figure 6, the 3-level hierarchical composite state goal model contains 11 states, 8 transitions, and 18 arcs. Two of the states are composite states. The one in top level is the goal of agent whereas the other one composite state in the middle level is sub goal. With the concurrent_to transition objects and jump_to transition objects, the composite state goal model can model complex problems that are difficult to model using existing goal models.

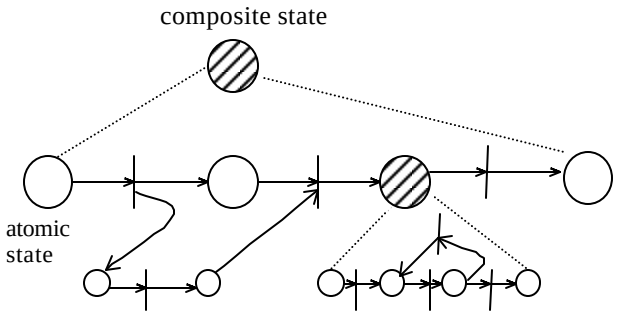

Figure 6 Composite state goal model

In the Goal-net, other than the basic types, such as ID, Description, and Status, S also includes type Duration to indicate the delay, type Distance to indicate state distance to the goal or sub goal, and type WorthValue to indicate the goal measurement.

Both state objects and transition objects have their own properties and behaviors. Besides *state Id* and *description*, a state object has a time counter of type Duration which records the *duration* of a state, a *local distance value* and a *global distance value* of type Distance to indicate how close the current state is to the state of the sub goal and final goal respectively. One of the state object's behaviors is that it can compute the *worth value* of type WorthValue through the utility function. The utility function of the state object gives a quantitative value for specifying the partial goal that the agent has reached. With the above measurements, an agent is able to choose its next sub-goal autonomously based on its available resources.

A transition object has *input states* and *output states*. It is associated with a task list and a *fire condition* function. The fire condition function specifies the fire condition of progress from the input states of a transition to its output states. The most important behavior of a transition is when the fire condition is satisfied, it can *fire* to remove the tokens from its input states and add tokens to its output

states. Therefore, the evaluation of the fire function enables the agent to decide its behaviors autonomously.

Apart from defining the states and the transitions, the model also defines a set of firing rules, which provides a mechanism for capturing and denoting dynamic characteristics of the goal model. In particular, firing rules of the model can be summarized as follows:

1) A transition is enabled when all the input states are active, i.e. hold tokens;
2) A transition fires as soon as the agent has successfully carried out the tasks specified in the task list;
3) When a transition fires, its input states become inactive, and its output states are activated;
4) A jump_to transition is enabled when certain conditions are met;

The goal pursuing of an agent starts from the top state which represents the goal of the agent; then goes through the hierarchical model; and finally goes back to the top state. The goal of the agent is said to be reached at this time. Then the agent will start from the top state again for the next time pursuing.

With different combination of direct_to, concurrent_with and jump_to transitions, a wide range of complicated relationships between states can be represented, which could accommodate various complex goals of the intelligent forecasting agents.

4. GOAL-BASED INTELLIGENT AGENT MODEL

The composite state goal model defines intelligent behavior of an agent. States represent the goal, sub goals and transit states. Transitions represent the dynamic behaviors. The tasks being fulfilled in each transition consist of functionalities of the agent application, which include agent basic functions and application specific functions. The goal-based intelligent agent model is presented in Figure 7.

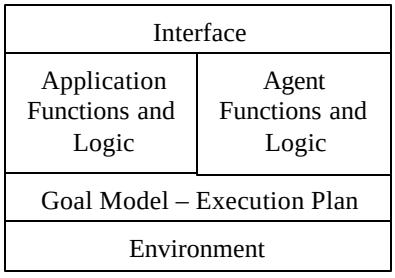

Figure 7 The goal-based intelligent agent model

In this model, the environment layer defines the agent running environment including system environment (system architecture, operating system, network, knowledge base, etc), communication mechanism (communication method, communication language, etc) and agent development environment (development language, agent framework, etc).

The goal-model layer defines the goal model of agent. The goal model is derived based on the business processes, work flow and business logic of the application. The goal model will become the execution plan of agent when the agent is running.

The function layer includes application functions and logic component and agent function and logic component. Application functions and logic component consists of application specific functions and business logic. They are application dependent. Agent functions and logic component consists of agent basic functions and working logic. They are application independent. The transition objects will invoke both application functions and agent functions during task fulfillment according to the goal model.

The interface layer facilitates the agent to interact with users, other agents and the running environment. The agent communicates through the interface layer either by application functions or agent own functions.

5. CASE STUDY: AN INTELLIGENT BUSINESS FORECASTING AGENT

The composite state goal model provides an efficient means for denoting the complex goals of intelligent agents. An example is given in this section to illustrate how the goal of a business forecasting agent can be represented by the composite state goal model.

As shown in Figure 8, assuming that S_1, the composite object in the root of the hierarchical model, represents the overall goal of an intelligent forecasting agent:

S_1. id = 1, S_1. description = “forecasting”

The overall goal of the agent is split into a set of sub-goals connected via transitions:

S_{10}. id = 10, S_{10}. description = “initialized”
S_{11}. id = 11, S_{11}. description = “data collection”
S_{12}. id = 12, S_{12}. description = “data preparation”
S_{13}. id = 13, S_{13}. description = “forecasting model training”
S_{14}. id = 14, S_{14}. description = “forecasting result reasoning”
S_{15}. id = 15, S_{15}. description = “forecasted”

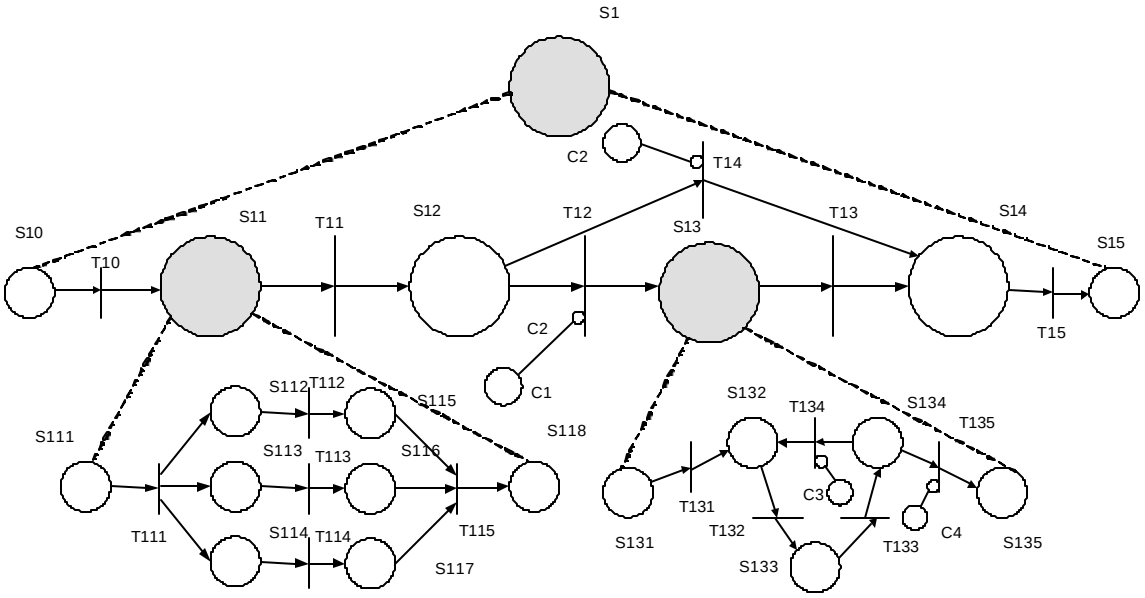

Figure 8 An example of composite state goal model

As shown in the level 2 of the hierarchical model, state S_{11} and S_{12} , S_{12} and S_{13} , S_{13} and S_{14} are connected by direct-to transitions, which indicates the agent can go through these states sequentially to reach the goal. A jump_to transition joins S_{12} and S_{14}, which implies that when certain conditions (C_2) are satisfied, the agent may jump to the S_{14} directly from S_{12}. For example, if the forecasting model has been trained, after the data is well prepared, agent can make inference without re-training the model.

Similarly the sub goals can be further split into a set of state objects connected via transitions. For instance, S_{11} is split into a set of state objects {S_{111}, S_{112}, S_{113}, S_{114}, S_{115}, S_{116}, S_{117}, S_{118}}:

S_{111}. id = 111, S_{111}. description = “initialized”
S_{112}. id = 112, S_{112}. description = “collecting data from source A”
S_{113}. id = 113, S_{113}. description = “collecting data from source B”
S_{114}. id = 114, S_{114}. description = “collecting data from source C”
S_{115}. id = 115, S_{115}. description = “new data collected from source A”
S_{116}. id = 116, S_{116}. description = “new data collected from source B”
S_{117}. id = 117, S_{117}. description = “new data collected from source C”
S_{118}. id = 118, S_{118}. description = “new data collected”

Another composite state S_{13} is split into a set of state objects {S_{131}, S_{132}, S_{133}, S_{134}, S_{135}}:

S_{131}. id = 131, S_{131}. description = “initialized”
S_{132}. id = 132, S_{132}. description = “collecting data from database”
S_{133}. id = 133, S_{133}. description = “computing forecasting result”
S_{134}. id = 134, S_{134}. description = “checking error and adjusting parameters”
S_{135}. id = 135, S_{135}. description = “trained”

As shown in the level 3 of the hierarchical model, there are three concurrent_with transitions among S_{112} to S_{115}, S_{113} to S_{116} and S_{114} to S_{117} which imply the concurrent relationships among them i.e. the agent collects data from the data source A, B, C concurrently.

Some elements of the goal model CSGM are as following:

Σ = {ID, Description, Status, Duration, Distance, WorthValue};

S = {S_1, S_{10}, S_{11}, S_{12}, S_{13}, S_{14}, S_{15}, S_{111}, S_{112}, S_{113}, S_{114}, S_{115}, S_{116}, S_{117}, S_{118}, S_{131}, S_{132}, S_{133}, S_{134}, S_{135}};

T = {T_{10}, T_{11}, T_{12}, T_{13}, T_{14}, T_{15}, T_{111}, T_{112}, T_{113}, T_{114}, T_{115}, T_{131}, T_{132}, T_{133}, T_{134}, T_{135}};

A = {$S_{10}ToT_{10}$, $T_{10}ToS_{11}$, $S_{11}ToT_{11}$, $T_{11}ToS_{12}$, $S_{12}ToT_{12}$, $T_{12}ToS_{13}$, $S_{13}ToT_{13}$, $T_{13}ToS_{14}$, $S_{12}ToT_{14}$, $T_{14}ToS_{14}$, $S_{111}ToT_{111}$, $T_{111}ToS_{112}$, $S_{112}ToT_{112}$, $T_{111}ToS_{113}$, $S_{113}ToT_{113}$, $T_{111}ToS_{114}$, $S_{114}ToT_{114}$, $T_{112}ToS_{115}$, $S_{115}ToT_{115}$, $T_{113}ToS_{116}$, $S_{116}ToT_{115}$, $T_{114}ToS_{117}$, $S_{117}ToT_{115}$, $T_{115}ToS_{118}$, $S_{131}ToT_{131}$, $T_{131}ToS_{132}$, $S_{132}ToT_{132}$, $T_{132}ToS_{133}$, $S_{133}ToT_{133}$, $T_{133}ToS_{134}$, $S_{134}ToT_{134}$, $T_{134}ToS_{132}$, $S_{134}ToT_{135}$, $T_{135}ToS_{135}$};

D = {S_1, S_{11}, S_{13}};

N = {$S_1S_{10}S_{15}$, $S_{11}S_{111}S_{118}$, $S_{13}S_{131}S_{135}$};

$B(S_1) = S_1S_{10}S_{15}$;

$B(S_{11}) = S_{11}S_{111}S_{118}$;

$B(S_{13}) = S_{13}S_{131}S_{135}$;

B(others) = 0;

H = 3;

Other elements of the goal model need to be decided in real application design.

As we can see, the goal and the sub-goals of the forecasting agent for carrying out various business forecasting processes, such as data collection, data preparation, model training, reasoning etc., can be well modeled by the composite state model. Moreover, as each state also has a worth value and a distance value to the sub goal and final goal, the partial goal that the agent has reached at each stage can be quantitatively measured. The measurement of the partial goal also enables the agent to evaluate, report its progress and choose the next sub goal autonomously for reaching its final goal.

6. EXPERIMENTAL RESULTS AND DISCUSSIONS

A prototype of intelligent business forecasting agents has been developed. It is based on the proposed goal model, goal-based agent model and our earlier work on business forecasting agent framework [7]. The agent is constructed using the Java based agent toolkit we developed earlier for building business forecasting agents.

In order to design and implement the proposed goal-based agent model, we modified the business forecasting agent framework [7] accordingly. The proposed composite state goal model is designed in the process unit of the framework and model data will be stored in the knowledge base. The control unit will read the goal model data from the knowledge base and interpret them to conduct the execution of the agent. The business forecasting model is still implemented by the knowledge unit and the model data and parameters are stored in the knowledge base as before. All the application functions and agent functions are defined in the action unit.

The experiment of the prototype system simulates an example case for forecasting the exchange rate of US dollar to Singapore dollar. We created three simple web pages to represent the three data sources respectively. Each page contains one set of forecasting data. A program was running in the background updating the three web pages every 30 minutes using 19 testing data sets. The forecasting agent monitored and checked the web pages at a time interval of 20 minutes.

The agent collected the data from the web pages through network connection autonomously. It converted the collected data whenever the new data arrived. The agent transformed a fuzzy neural network based business forecasting model as its knowledge. It trained the forecasting model based on the most recent training data and generated the forecasting results using reasoning algorithms of the forecasting model.

Two agents were evaluated for forecasting the exchange rate of Singapore dollar and US dollar respectively. One agent *A* used the goal model without time limit, another agent *B* used the goal model with a time limit. We found that two agents were working well. When the time limit of agent *B* was shorten enough, the agent skipped the training step and jumped to forecasting step directly. This is reasonable because it is not necessary to train the model every time.

From the experiment, we obtained the following results:

1. The composite goal model can well model the complex goals of business forecasting agents; the use of goal measurement improves the capability of the intelligent business forecasting agents.
2. The traditional goal model cannot be suitable for the complex business forecasting system. In the case of Agent B, if using the traditional goal model, when the time limit is reached, the agent would simply stop at the training stage even though it is not necessary to train the forecasting model every time.
3. The goal-based agent model is easily designed and implemented. Since the application functions and logic can be designed in one component of the model and the goal model data is stored in knowledge base, which means the goal model can be dynamically generated during runtime using the goal model data, the reusability of the goal-based agent model is very high.
4. Compared with our previous work presented in [7] which was developed based on existing goal model, task oriented goal model, the upgraded version of the agent framework, which was developed based on the proposed composite state goal model, can implement dynamic agent behaviors to adapt the real business forecasting environment.

7. CONCLUSION

In this paper, we have presented a practical new approach to modeling the agent goals. The overall goal and sub goals of the agent can be represented by composite state objects in different level of the hierarchical structure of the model. The composite state objects can be further split into a set of state objects connected via transitions. The firing rules of the transition and the goal measurements facilitate the dynamic characteristics of the agent's goals. The goal-based agent model is also proposed in this paper for construct agents using the goal model. Compared with the existing goal models, the composite state goal model has the following advantages:

- The composite state goal model decomposes a complex goal of intelligent agent into sub goals by which the modeling complexity is reduced;
- By modeling the goal using the composite state goal model, the agents are able to present not only behavior autonomy but also goal autonomy;
- The composite state goal model supports partial goal by which the flexibility of the agent is increased;
- The quantified goal measurement enables the agent to reach its goal efficiently.

It has been shown that the agent model proposed in this paper not only overcomes the drawbacks of existing goal models but also enables agents to present both behavior autonomy and goal autonomy. An illustration of the practice of using the goal model is also given by the case study which also demonstrates the proposed goal model can well model the complex goals of business forecasting agents for managing various business forecasting processes in a whole life cycle of business forecasting.

REFERENCES

1. H. Nwana and D. Ndumu, "A Perspective on Software Agents Research", *Knowledge Engineering Review*, 14(2), 1, 1999.
2. J. S. Armstrong, "Standards and Practices for Forecasting", *Principles of Forecasting: A Handbook for Researchers and Practitioners*, Kluwer Academic Publishers, Norwell, MA, 2000.
3. B. Küsters, "The Forecasting Report: A comparative survey of commercial forecasting systems", *IT Research*, November, 1999.
4. Z. Q. Shen and R. Gay, "Agent Oriented Business Forecasting", *The Pacific Asian Conference on Intelligent System 2001*, Seoul, Korea, November 16-17, 2001.
5. J. S. Rosenschein and G. Zlotkin, "Rules of Encounter: Designing Conventions for Automated Negotiation among Computers", MIT Press, Cambridge, Massachusetts, USA, 1994.
6. J. L. Peterson, *Petri Net Theory and the Modeling of Systems*, Prentice-Hall, N.J., 1981.
7. Z. Q. Shen, R. Gay, X. Li and Z. H. Yang, "An Agent Approach for Intelligent Business Forecasting", *Intelligent Systems: Technology and Applications*. Leondes, C. T., Ed., CRC Press LLC, 2002.