A Method for Modeling Service Management of e-Learning

JYHJONG LIN
Department of Information Management, Ming Chuan University
Kweishan, Taoyuan County,
TAIWAN 333
E-mail: jlin@mcu.edu.tw, Fax: 886-3-3593875

Abstract: - For the rapid advances of Internet technologies and Web applications in recent years, providing opportunities to learn outside of the traditional classroom-based education has gained many attentions as a new theme for prospect learners to acquire knowledge in a more convenient way. In this new paradigm of the so called electronic learning (e-Learning), many efforts have been made to build Web based learning systems that manage desired e-Learning processes. From the managerial perspective on education, this means that each desired e-Learning process is monitored and controlled for fulfilling an expected learning objective. In this paper, we propose an object-oriented modeling method that addresses this issue by dividing required mechanisms into three layers: learning objective, learning service agent, and learning service composition ones. With this architecture, e-Learning processes are managed via the recognition of a learning objective, the employment of a learning service agent that arranges a process of demanded learning services for achieving the objective, and the confirmation of interactions/coordination among these services in achieving the objective. For specification, an object-oriented model is presented for each layer that describes the working detail of that layer. To illustrate, these models are applied in the fulfillment of an e-Learning plan for learning about Software Engineering that involves a set of learning objectives to be achieved by various processes of learning services.

Key-Words: - e-Learning, service management, object-orientation, modeling method

1 Introduction

Conceptual modeling is an important technique for representing (part of) a complex situation in an abstract manner with concise notations. It has been commonly used, for example, in analyzing and specifying user requirements of a computer-based application, as well as collecting and representing information required for dealing with complex technical and/or managerial issues to be resolved. In general, conceptual modeling can be achieved by using function-, data-, or object-oriented ways where the development of object-oriented ones is particularly motivated by the drawbacks and problems in the other two kinds: the significant features and benefits of object-oriented approaches would make resultant models more abstract and thus easier to be understood, maintained, and reused.

For the rapid advances of Internet technologies and Web applications in recent years, providing opportunities to learn outside of the traditional classroom-based education has gained many attentions as a new theme for prospect learners to acquire knowledge in a more convenient way. In this new paradigm of the so called electronic learning (e-Learning), many efforts have been made to build Web based learning systems that manage desired e-Learning processes. From the managerial perspective on education, this means that each desired e-Learning process is monitored and controlled for fulfilling an expected learning objective (or goal used interchangeably in the literature [1]).

In our knowledge, this managerial issue is needed in order to specifically deal with those many dynamic and complicated concepts in the e-Learning paradigm, including instructor, learner, learning material, learning activity/ process, and learning collaboration/coordination. As stated above, in order to address this complex issue with an abstract conceptual modeling mechanism, it is not uncommon to think of the powerful object-oriented techniques that possess such features as encapsulated specifics of an object and interacted/ coordinated nature of its behaviors with other objects; these features make an object-oriented approach easier to be configured for an extensive support of addressing this issue. To account for this, we propose in this paper such an object-oriented method for modeling and specification of the managerial issue of e-Learning.

In our knowledge, e-Learning is managed to concern what learners really care about that includes the recognition of a learning objective and how the objective is specified and achieved by required learning services under a commitment mechanism (i.e., engaging the achievement of this objective through the provision of a designated process of these learning services). Many approaches that deal with (part of) these needs have been presented; most of which focus mainly on specifying/directing required learning services for accomplishing a process of learning activities, including for instance (1) rule-based systems [2] that allow by means of access rules specifying and directing a specific process of learning activities, (2) relationship based systems [3] that use logic relationships to define a course structure with the relationships among its containing course components; and (3) workflow based systems [4-6] that employ the power of workflows to define a stream of activities that constitute a learning process.

As one may notice, nevertheless, these approaches do not look at a learning process from the managerial perspective of education. That is, a learning process is defined in that it is intended to accomplish some learning objectives; conversely, a learning objective may be achieved by some learning processes. In general, although these approaches support well the provision of learning services for accomplishing a process of learning activities, they have the following deficiencies: (1) their mechanisms describe how services collaborate, e.g., being sequenced and coordinated with each other, in a rather statically structured manner such that the relationships among services cannot be easily extended/modified for reusing these services in achieving various but possibly related learning objectives; and (2) as stated above, they do not consider the necessities of specifying learning objectives that commonly are recognized first by the learners and identified then to be achieved by required learning services; further, how these learning objectives are specified is critical such that the possible relationships, e.g., extensions, combinations, and associations, among learning objectives can be easily maintained for reusing these objectives in dealing with different learning situations; in our view, making these relationships maintainable would specifically benefit for adapting to a learner's needs by easy adjustment, e.g., extensions or modifications, of his/her learning objectives to respond to the dynamic and changeable learning environment nowadays.

To overcome these limitations, our approach takes advantage of the object-oriented paradigm, together with the use of visual notations and formal mechanisms, to specify learning objectives and their corresponding service-level collaborations. It employs three layers of constructs: learning objective, learning service agent, and learning service composition ones; with this architecture, the management of learning services for a prospect learner is accomplished by recognizing a set of related learning objectives where each objective is engaged by a learning service agent that arranges a composition of learning services for achieving the objective. For specification, an object- oriented model is presented for each layer that describes the working detail of that layer: (1) a learning objective model that specifies the desired learning objectives and their relationships; (2) a learning service agent model that presents the agents responsible for these objectives and the compositions of learning services these agents arrange for achieving these objectives; and (3) a learning service composition model that describes the compositions and interactions among those learning services in a composition.

With these three models, our specifications start from a higher-level of learning objective descriptions and end at a lower-level of learning service compositions. Note that our service composition model imposes formal constructs based on Petri nets [7-9] such that verifying objectives of the service compositions can be conducted; we believe this formality is very important for the managerial purpose on education, since what learners really care about is the achievement of objectives by demanded learning services. For illustration, the three models are applied in the fulfillment of an e-Learning plan for learning about Software Engineering that involves a set of learning objectives to be achieved by various processes of learning services.

This paper is organized as follows. Section 2 overviews the background and motivation of the proposed approach. Section 3 presents our three models. The method that provides the guidance on how these models are applied in a step by step manner will be presented in Section 4. Finally, Section 5 has conclusions and future work.

2 Background and motivation

In e-Learning environments [10,11], learning activities should be supported by respective learning services and monitored for ensuring their achievement on desired learning purposes. In this context, some approaches have been presented that define specific ways for specifying/directing required learning services in order to accomplish a process of learning activities. These approaches include (1) rule-based systems [2] that allows by means of access rules specifying and directing a specific process of learning activities, (2) relationship based systems [3] that use logic relationships to define a course structure with the relationships among its containing course components; and (3) workflow based systems [4-6] that employ the power of workflows to define a stream of activities that constitute a learning process. As one may notice, nevertheless, these approaches do not look at a learning process from the managerial perspective on education. That is, a learning process is defined in that it is intended to achieve some learning objectives; conversely, a learning objective may be achieved by some learning processes.

In general, these approaches support well the provision of learning services for accomplishing a process of learning activities; they however have the following deficiencies: (1) their mechanisms describe how services collaborate, e.g., being sequenced and coordinated with each other, in a rather statically structured manner such that the relationships among services cannot be easily extended/modified for reusing these services in achieving various but possibly related learning objectives; and (2) as stated above, they do not consider the necessities of specifying learning objectives that commonly are recognized first by the learners and identified then to be achieved by required learning services; further, how these learning objectives are specified is critical such that the possible relationships, e.g., extensions, combinations, and associations, among learning objectives can be easily maintained for reusing these objectives in dealing with different learning situations; in our view, making these relationships maintainable would specifically benefit for adapting to a learner's needs by easy adjustment, e.g., extensions or modifications, of his/her learning objectives to respond to the dynamic and changeable learning environment nowadays.

Our method is proposed to supplement those deficiencies in current approaches by providing a visual formalism for easy specification and maintenance of learning objectives and their corresponding service compositions. To address the complexity of required mechanisms, it supports the specification in a top-down fashion. As results, a higher-level learning objective model is created first that describes desired learning objectives and their possible relationships without considering detailed specification. That is, the detailed specification via learning service agent and learning service composition models starts after all related learning objectives have been described in an abstract level. We think this provides better understanding about critical objectives before proceeding too early to formally specify their accomplishments using some complex notations. Finally, due to its formal semantics of the learning service composition model, behavioral verification of satisfying the desired objectives can be conducted via formal analysis of the model [12]. Note that due to its enhanced modeling constructs for an extensive support of the objective, agent, and composition issues, our object-oriented model is different from other existing ones, including the most well-known UML [13-15]. Although these models can also be modified/extended to support the same specification as ours does, for space limitations, we do not address herein how such modifications/extensions may be conducted.

3 Modeling constructs

The modeling constructs of our approach include three models: (1) a learning objective model that specifies the desired learning objectives for a prospect learner and their possible relationships; (2) a learning service agent model that presents the agents responsible for these objectives and the compositions of learning services they arrange for achieving these objectives; and (3) a learning service composition model that describes the compositions and interactions among those learning services within/between a composition.

3.1 The learning objective model

In the literature, many classifications for objectives have been proposed as those discussed in [1] where a distinct is made between soft (non-functional) ones whose satisfaction cannot be established in a clear- cut sense and hard (functional) ones whose satisfaction can be established through verification techniques. Among other types of classification, in our knowledge, this distinct is most often referenced such that our model focuses on the specification of

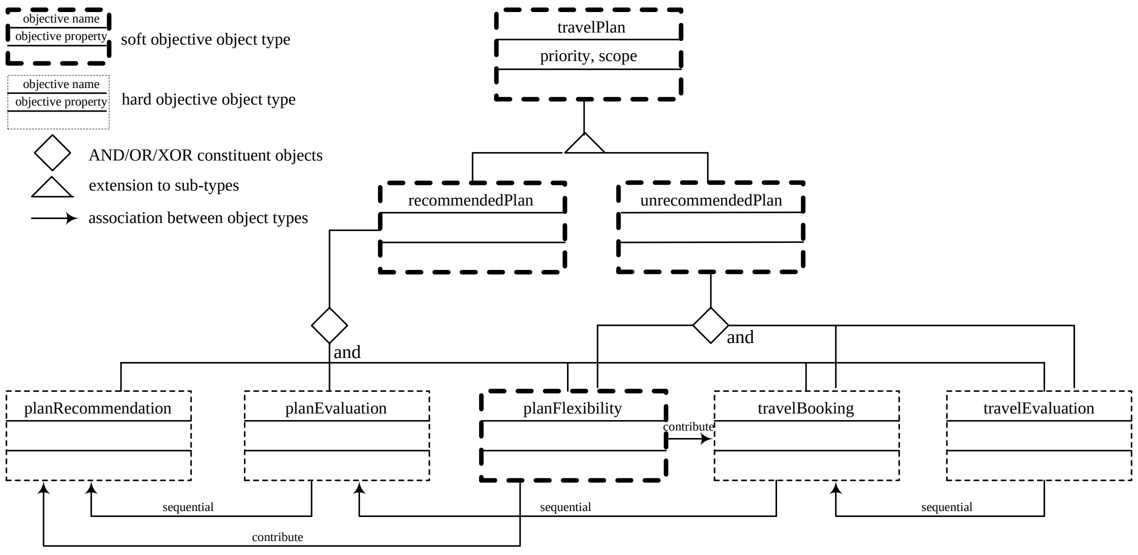

Figure 1: desired sub-objectives for a travel plan objective

learning objectives with soft and hard object types (classes). Figure 1 shows an example model that specifies by proper object types a 'Software Engineering Learning' objective that is extended as 'guided' and 'unguided' ones: to say, a prospect learner would learn about Software Engineering through either (1) a computer- guided sequence of learning sub- objectives: 'being briefed of learning materials', 'being suggested of lecturing processes', 'learning of materials under a selected process', and finally 'being assessed of learning effects', or (2) an unguided way: 'being briefed of learning materials', 'self-organizing of a preferred lecturing process', and finally 'learning of materials under the preferred process'. In these two ways, however, keeping adjustable on the 'being suggested of lecturing processes' and 'self-organizing of a preferred lecturing process' sub-objectives (i.e., be able to adjust the suggested/ self-organized processes) is an advanced sub- objective for making the learner more satisfied. As shown in the figure, a (soft or hard) (sub)-objective object is specified with (1) attributes such as objective priority and scope; (2) extensions into more specialized sub-types, or compositions with AND/OR/ XOR constituent objects [16,17]; and (3) associations with other (sub)-objective objects [18] such as 'sequential' that denotes an achievement sequence from *source* to *destination*, and 'contribute' that denotes an achievement contribution from *source* toward *destination*. Further, it is noticed that an object containing one or more constituent soft objects is specifically classified as a soft one; this is because a (sub)-objective comprising one or more constituent soft sub-objectives should be classified as a soft one due to its satisfaction depending on those of these constituent sub-objectives.

3.2 The learning service agent model

With a learning objective model, the learning service agent model is used to specify more detail about the desired agents that arrange demanded learning services for achieving those (sub)-objectives specified (note that the reader is referred to [19-21] for employing agents for the achievement of objectives). As shown in the figure, its description for each agent includes (1) attributes such as the effective period of its responsibility and the resources required for its arrangement; (2) compositions of AND/OR/XOR constituent agents; (3) associations with other agents such as 'sequential' that denotes a service-providing sequence from *source* to *destination*; and (4) compositions of arranged learning services and how these services participate in achieving the (sub)- objective (i.e., with AND/OR/XOR relationships).
The modeling constructs of the learning service agent model include four kinds of object type: soft/hard objective, agent, and service ones. In particular, each agent object is specified for realizing a desired agent that arranges a composition of learning services for achieving a soft/hard (sub)-objective; its specification includes a name, required properties (e.g., the resources accessed), and a set of operations that are purposed for engaging the achievement of the (sub)-objective through invoking the operations of its constituent agent/service objects (that is, in our means, the execution of each operation would result in those of the operations in constituent agent/service objects that collaboratively produce a final result as the output of the operation). In turn,

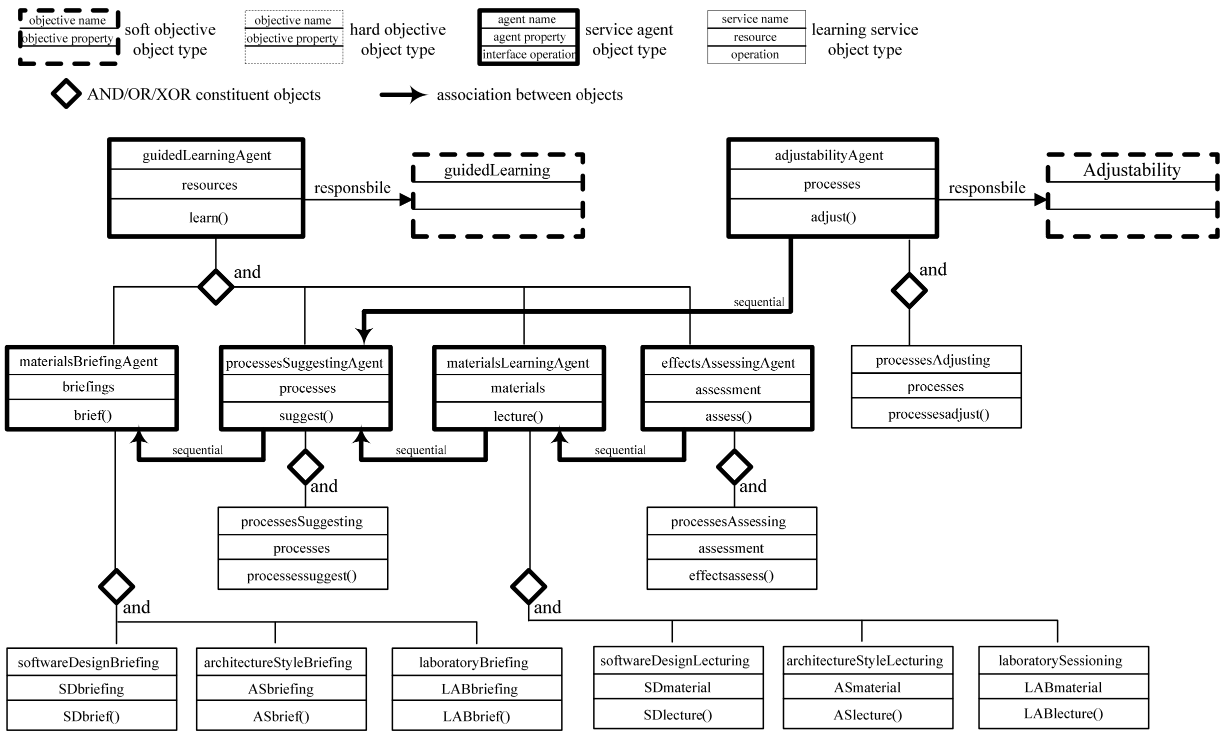

Figure 2: agents responsible for achieving the objectives

each constituent service object is specified for modeling a learning service demanded for achieving a (sub)-objective with a description about its operations and accessed resources.

As shown in Figure 2, two agents are identified that are responsible for achieving respectively the two ‘guidedLearning’ and ‘Adjustability’ sub-objectives under the ‘Software Engineering Learning’ one identified in Figure 1. Further, the ‘guidedLearningAgent’ agent contains (requires all of the) four constituent ones that are responsible for achieving respectively the four constituent sub-objectives of the ‘guidedLearning’ one. Specifically, the ‘guidedLearningAgent’ agent object is specified with two ‘effective period’ and ‘resources’ properties and a ‘learn()’ operation for achieving the ‘guidedLearning’ sub-objective. For the ‘learn()’ operation, in particular, the starting of its execution would result in the executions of some operations in the four constituent agent objects, and the executions of these constituent operations would end before the ending of its execution. The execution sequence of these constituent operations is specified with the ‘sequential’ associations between the four constituent agent objects. Finally, for each constituent agent object, the services it arranges are specified with prospect service objects between which an ‘and’ relationship is identified. This simply means that all service objects are required and their operations shall be executed under some designated execution sequences. For simplicity in our model, however, the specification of these execution sequences will be presented in the service composition model below.

3.3 The learning service composition model

With a learning service agent model, the learning service composition model is finally used to present in detail how the operations of an agent object engage the achievement of a (sub)-objective by invoking those of its constituent service objects that collaborate through various sequences, e.g., sequential, parallel, and exclusive. In general, its modeling constructs are based on Petri nets [7-9] with a set of (normal/control) transitions and places. Normal transitions specify the operations that are executed for achieving desired (sub)-objectives, while control transitions impose the control flows for those executions of normal transitions. Likewise, places are divided into two kinds: normal places that hold entity objects for the executions of transitions, and control places that hold control objects for controlling the executions of transitions (e.g., a ‘s’ object for employing a sequential execution, a ‘x’ object for an exclusive one, and a ‘p’ object for a parallel one). Each transition is specified with a name and a set of interaction places that its execution accesses. With this specification, a transition is executable

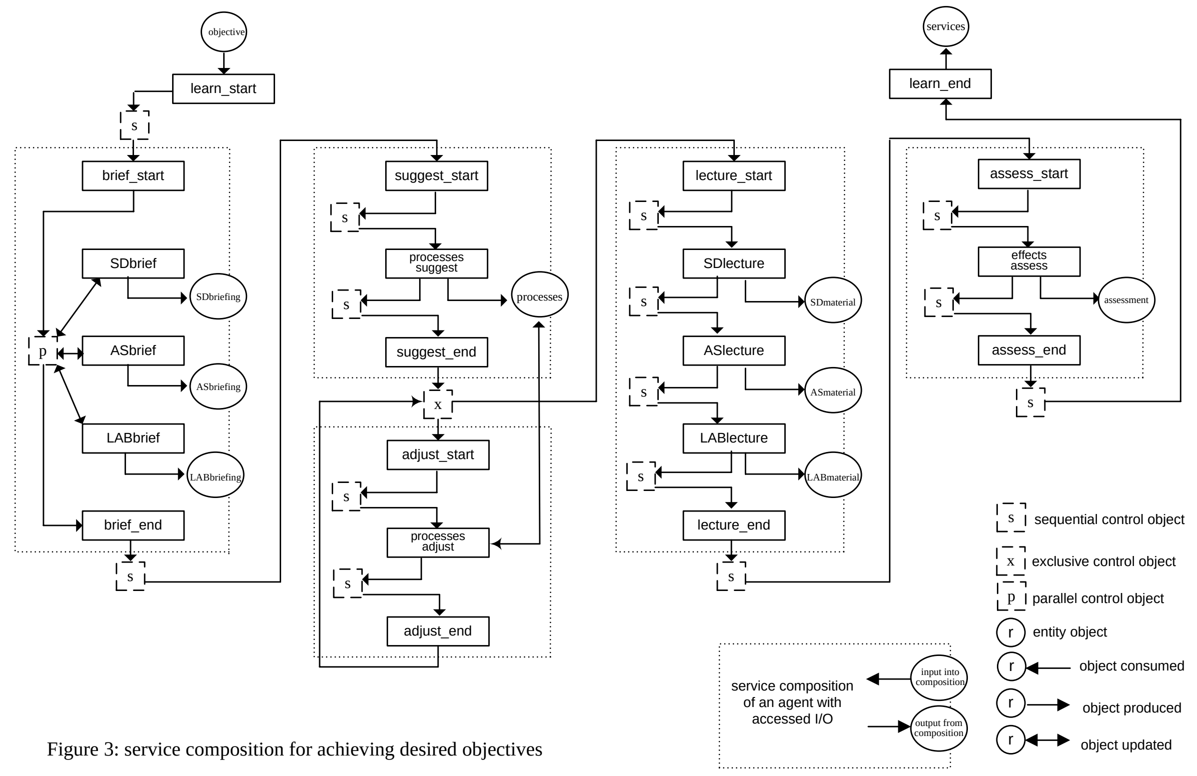

Figure 3: service composition for achieving desired objectives

if and only if each of its input places contains an object, and once executed, objects in its input places are consumed by the transition and objects in its output places are produced respectively.

In Figure 3, a service composition model is presented that describes in detail how the execution of the ‘learn()’ operation in the ‘guidedLearningAgent’ agent object results in those of the operations in four constituent agent objects, and also how the execution of each operation in a constituent agent object results in those of the operations in respective constituent service objects. As shown in the figure, at the starting of the execution of the ‘learn()’ operation, a ‘s’ control object is produced for imposing a sequential execution of the ‘brief()’ operation in the ‘materialsBriefingAgent’ agent object, while in turn the execution of the ‘brief()’ operation results in a ‘p’ control object produced for imposing a parallel executions of three operations in respective service objects. It is noticed that for those outputs produced by the three service operations, they collectively form the output of the ‘brief()’ operation. Further, after the execution of the ‘suggest()’ operation in the ‘processesSuggesting Agent’ agent object, a ‘x’ control object is produced for imposing an exclusive execution between the ‘adust()’ operation in the ‘adjustabilityAgent’ agent object and the ‘lecture()’ operation in the ‘materials LearningAgent’ agent object. This is particularly specified for illustrating the situation that after being suggested possible lecturing processes, a learner may either select one of them for starting his/her study journey or adjust some of these processes before selecting one as his/her study journey.

Finally, with the learning service composition model, one may see that since the model is based on Petri nets, its formal semantics can then be applied for behavioral verification of how the ‘learn()’ operation in the ‘guidedLearningAgent’ agent object engages the achievement of the ‘guidedLearning’ sub-objective by a collaboration of the four constituent agent operations where each of which in turn results in that of respective constituent service operations. This can be achieved via decision procedures that traverse the reachability graph derived from the agent/service composition. The reader is referred to [12] for more detail about this issue.

4 The modeling method

4.1 Specifying the learning objective model

In our method, the specification of learning objectives starts from depicting a learning objective and how it may be extended into sub-types or be composed of AND/OR/XOR constituent sub-objectives with the following steps:

1. Start with describing a desirable learning objective by specifying an objective object.

The possible extensions and/or compositions of the object are then explored to identify what sub-types the objective may be extended into and/or what constituent sub-objectives may have (with an AND/OR/ XOR relationship among each other) that compose the objective. Then, based on whether the sub- objective it specifies is soft or hard, each identified constituent object is correspondingly classified as a soft or hard one. Finally, in case one or more constituent objects are soft ones, the objective object is classified also as a soft one.

For our example shown in Figure 1, a 'Software Engineering Learning' objective is identified and specified as an objective object that is extended into two sub-types for specifying the two more specific ('guided' and 'un-guided') sub-objectives. Further, each of the two extensions is composed of a set of constituent objects with an AND relationship among each other; also, since one of these constituent objects is a soft one, the extension itself is classified as a soft one.

2. With (sub)-objective objects and their extensions/ constituents identified, the attributes of these objects such as priority and scope are then specified. Also, check if any associations between two objects are needed. For example, a 'sequential' association may be imposed that denotes the achievement of a (sub)-objective is earlier than that of the other one; also, a 'contribute' association denotes the achievement of a (sub)-objective contributes toward that of the other one.

   As shown in Figure 1, three 'sequential' associations are specified among four hard sub-objective objects to denote an achievement sequence among four relevant sub-objectives. In addition, a 'contribute' association is imposed that denotes an achievement of the 'Adjustability' sub-objective contributes toward those of the 'processesSuggesting' and 'processOrganizing' sub-objectives. Note that in our view, since (sub)-objectives are achieved by demanded learning services, any associations between two (sub)-objective objects may result in corresponding relationships between those respective service objects; for example, the 'contribute' association from 'Adjustability' to 'processesSuggesting' may result in some service objects for achieving 'Adjustability' has the same association with those service objects for achieving 'processesSuggesting' and hence participates in achieving 'guidedLearning'. As presented earlier in our spec. constructs, such possible relationships are specified in detail in the learning service composition model.

4.2 Specifying the learning service entity model

With a learning objective model, our method advocates the specification of a learning service entity model that presents more detail about what a composition of learning services are demanded that be arranged by a specific entity for achieving a (sub)-objective. Therefore, a learning service entity is identified to arrange those learning services demanded for achieving a (sub)-objective with the following steps:

1. For each learning (sub)-objective identified, specify a learning service entity responsible for achieving it. Then, based on the identified compositions and associations between (sub)-objectives, consider possible compositions and associations between respective entities; as one may expect, these entities have usually the same relationships between each other as those between their corresponding (sub)-objectives. Afterwards, for each lowest-level entity (without constituent ones), identify what learning services are demanded in order to achieve the (sub)-objective it is responsible. With all entities and demanded learning services recognized, specify corresponding entity and service objects for describing these learning services that would be arranged by each entity object as in a composition. Finally, after employing respective entity objects for all identified learning (sub)-objectives and their constituent entity/service objects, how specific service objects participate in each composition (i.e., a service object may be employed for achieving more than one (sub)-objective) is also explored.

   As shown in Figure 2 that follows Figure 1, two entities are employed for achieving respectively the 'guidedLearning' and 'Adjustability' sub-objectives under the 'Software Engineering Learning' one identified in Figure 1. Further, the 'guidedLearningEntity' entity contains (requires all of the) four constituent ones that are responsible for achieving respectively the four constituent sub-objectives of the 'guidedLearning' one. Among these entities, 'sequential' associations are specified to show their service-providing sequences. In addition, a composition of three learning services are demanded for both of the 'materialsBriefingEntity' and 'materialsLecturingEntity' entities in order to achieve the 'materialsBriefing' and 'materialsLecturing' sub-objectives respectively (i.e., being briefed and lectured of learning materials about software engineering). For these entities and learning services, corresponding entity and service objects are then specified where the service for adjusting suggested processes participates also in the composition for achieving the 'guidedLearning' sub-objective (i.e., keeping adjustability is a commonly desired objective for a learner that allows him/her to adjust those processes suggested).

2. For each entity object responsible for a (sub)-objective, specify first its properties (e.g., the effective period of its responsibility and the resources required for its arrangement) and operations that are purposed for achieving the (sub)-objective. Then, for each of the service objects it arranges, identify what operations are required for practically providing required services as well as what resources are necessarily accessed during the services provision. As mentioned in our specification constructs earlier, the operations in a constituent entity/service object would be executed during the executions of some operations in its containing entity object in order to achieve the (sub)-objective addressed.

   As shown in Figure 2, the 'guidedLearningEntity' entity object responsible for the 'guidedLearning' sub-objective is specified with two 'effective period' and 'resources' properties and a 'learn()' operation. For the 'learn()' operation, in particular, the starting of its execution would result in the executions of some operations in the four constituent entity objects, and the executions of these constituent operations would end before the ending of its execution. Similarly, the 'adjust()' operation in the 'adjustabilityEntity' entity object would result in the execution of the 'processesadjust()' operation in the 'processesAdjusting' service object.

4.3 Specifying the learning service composition model

With a learning service entity model, the specification of a learning service composition model is then considered that presents in detail how constituent entity/service operations are executed during the execution of an operation in a containing entity object
and how they collaborate through various sequences (e.g., sequential, parallel, and exclusive) to achieve the (sub)-objective addressed.

1. For each operation of a containing entity object, identify what constituent entity/service operations would be involved for its execution and among them which one should be invoked first as an initialization of the involvement. Then, map each of these constituent operations into a corresponding normal transition, and further map the containing operation into two (i.e., start and end) normal transitions.

   As shown in Figure 3 that follows Figure 2, the 'learn()' operation in the 'guidedLearningEntity' entity object gets totally four constituent entity operations involved for its execution where the 'brief()' operation in the 'materialsBriefingEntity' entity object is first invoked to initialize this involvement. Also, it can be found that since the 'materialsBriefingEntity' entity object is a containing one, its 'brief()' operation gets totally three constituent service operations involved for its execution where these three service operations are possibly invoked in parallel during this involvement. Finally, the three constituent service operations are mapped into respective normal transitions, while the two containing operations are each mapped into two 'start' and 'end' normal transitions.

2. With normal transitions identified, the normal places that hold entity objects to be accessed by these transitions are then explored. In general, this is done by examining the accessed resources of the entity/service objects in which those operations mapped reside where these accessed resources are specified as distinct entity objects to be held in respective input/output normal places. Thereafter, consider the possible merging between any pair of output and input places in case their holdings present the same entity. Finally, with normal transitions and their accessed normal places determined, the execution controls of these transitions are identified that address their execution sequences for accomplishing the executions of any concerned containing operations. It is noted that, for our illustrative purposes herein, an execution sequence can proceed with three kinds of control:
sequential, parallel, and exclusive where specific control transitions and/or places that hold control objects are necessarily employed for realizing each of them. It is noticed that for each 'start' transition that initializes an execution sequence for accomplishing the execution of a containing operation, ensure if its output is sufficient for initializing the execution sequence; meanwhile, for the 'end' transition that follows the sequence, ensure if its input is produced properly after the ending of the sequence.

   As an example in Figure 3, the three normal transitions mapped for the execution of the 'brief()' operation are specified with their execution sequence entering a parallel way by the access of a 'parallel' control object produced by the 'brief_start' transition; the 'brief_end' transition has the control object consumed after the ending of the sequence. In turn, the execution of the 'brief()' operation as a whole is followed by that of the 'suggest()' operation (with a 's' control object) which is then followed in an exclusive way by the two 'adjust()' and 'lecture()' operations (with a 'exclusive' control object).

3. Finally, for each execution sequence that accomplishes the execution of a containing operation in an entity object, verify if it eventually produces the required resources for the containing operation in achieving the (sub)-objective addressed. Obviously, since each execution sequence is specified based on Petri nets, its formal semantics can be applied for this need via a decision procedure that traverses the reachability graph derived from the sequence [12].

As shown in Figure 3, four execution sequences are specified for accomplishing the execution of the 'learn()' operation in the 'guidedLearningEntity' entity object that engages the achievement of the 'guidedLearning' sub-objective. For each sequence, a pair of 'start' and 'end' transitions mapped for a containing operation are specified for its initialization and ending, and hence its reachability graph can be easily derived and traversed for ensuring the required resources produced for its corresponding containing operation.

5 Conclusions

Software requirements specification is a key activity in developing a computer-based application. Motivated by the problems in other methods, object-oriented specification methods are developed in order to produce software more understandable and maintainable. The method proposed in this paper is based on the object-oriented paradigm for formal specification about service management of e-Learning. In order to deal with the modeling complexity for the achievement of learning objectives by demanded learning services, learning (sub)-objectives, learning service agents, and learning services are identified and specified in a top-down fashion. As results, a higher-level learning objective model is created first that describes effectively desired learning (sub)-objectives and their possible relationships without considering detailed specification. That is, the detailed specification with service agent and composition models starts after all related learning (sub)-objectives have been described in an abstract level. We think this provides better understanding about desired learning (sub)-objectives before proceeding too early to formally specify their achievement using some complex notations. Finally, due to its formal semantics of the learning service composition model, behavioral verification of satisfying those desired (sub)-objectives can be conducted via formal analysis of the model.

The work for service management of e-Learning has already become a new discussion. Although some researches about it have been done, but none of them provides a complete mechanism for supporting all about a holistic view between (sub)-objectives and learning services, a flexible reusing of these (sub)- objectives and services, and a visual formalism for their specification. Our method presented herein provides an effort on these issues by using object-oriented visual models for specifying learning (sub)- objectives and their possible extensions and/or constituents, employing service agents for engaging the achievement of these (sub)- objectives, and imposing verifiable service compositions for achieving these (sub)-objectives under the arrangement of these service agents. In our knowledge, these models are much helpful for identifying and specifying those important requirements about learning (sub)-objectives and their achievement by demanded learning services.

As the technical issues about learning services are getting rapidly matured in these years, more learning services are expected to be available in the near future and hence a comprehensive mechanism for full supports of their management will certainly become much more desirable. Thus, the development of such a mechanism is a desired field. In our view, using object-oriented techniques together with sound modeling constructs is a promising approach for an effective construction of the mechanism. In our future work, we will explore further some other key issues that our models have not addressed yet, including effective registration and selection of learning services before creating a management-level session for learning services, and desired manipulations (e.g., create, delegate, assign, cancel, and release) on the session during its lifecycle. As stated in [19,20], these issues are critical for keeping a session flexible to achieve managerial purposes. Therefore, how to specify them by using our models' constructs will be carefully explored. Meanwhile, we will construct a tool to facilitate practical application of our models. These include a design environment for building the abstract learning objective model and then deriving the detailed learning service agent and composition models. The specification method presented in section 4 will be integrated with the tool when constructing the three models.

References:

[1] A. Lamsweerde, "Goal-Oriented Requirements Engineering: A Guided Tour," Proc. of 5th IEEE Int'l Conf. on Req. Eng., Aug. 2001, pp. 249-262.

[2] G. de Hesus Hoyos-Rivera, et al., "A Flexible Architecture for Collaborative Browsing," Proc. of 11th WETICE, 2002.

[3] A. Steinacker, te al., "Combining Semantic Networks with Metadata for Learning Resources to Build A Web Based Learning System," Proc. of ED-MEDIA, 2001.

[4] J. Lin, et al., "On Workflow Enabled e-Learning Services," Proc. of the Int'l Conference on Advanced Learning Technology (ICALT), 2001.

[5] M. Cesarini, et la., "Carrying on the e-Learning Process with A Workflow Management Engine," Proc. of ACM SAC, 2004, pp. 940-945.

[6] C. Lee, et al., "Analysis on the Adaptive Scaffolding Learning Path and the Learning Performance of e-Learning," WSEAS Trans. On Information Science and Applications, vol. 5, no. 4, 2008, pp. 320-330.

[7] J. Peterson, "Petri Nets," ACM Computer Surveys, vol. 9, no. 3, Sep. 1977, pp. 223-252.

[8] J. Peterson, Petri Net Theory and The Modeling of Systems, Prentice-Hall, 1981.

[9] E. Yiannis and L. Thomas, "Specification and Analysis of Parallel/Distributed Software and Systems by Petri Nets with Transition Enabling Function," IEEE TSE, v. 18, 1992, pp. 252-261.

[10] E. Verdú, et al., "An analysis of the Research on Adaptive Learning: The Next Generation of e-Learning," WSEAS Trans. on Info. Science and Applications, vol. 5, no. 6, 2008, pp. 859-868.

[11] S. Kim, et al., "A Design of u-Learning System based on Memory Theories in Ubiquitous Environments," WSEAS Trans. on Advances in Engineering Education, vol. 5, no. 2, 2008, pp. 77-82.

[12] J. Lin, et al., "Object-Oriented Specification and Formal Verification of Real-Time Systems," Annals of SE, 1996, vol. 2, pp. 161-198.

[13] G. Booch, et al., The Unified Modeling Language User Guide, Addison Wesley, 1999.

[14] M. Fowler and K. Scott, UML Distilled: Applying the Standard Object Modeling Language, 2nd Ed., Addison Wesley, 2000.

[15] J. Rumbaugh, et al., The Unified Modeling Language Reference Manual, 1999.

[16] A. Dardenne, et al., "Goal-Directed Concept Acquisition in Requirements Elicitation," Proc. of 6th Int'l Workshop on Software Specification and Design, 1991, pp. 14-21.

[17] A. Dardenne, et al., "Goal-Directed Requirements Acquisition," Science of Computer Programming, 1993, pp. 3-50.

[18] R. Darimont, et al., "GRAIL/KAOS: An Environment for Goal-Driven Req. Eng.," Proc. of 20th ICSE, 1998, pp. 58-62.

[19] A. van Lamsweerde, et al., "Managing Conflicts in Goal-Driven Req. Eng.," IEEE Trans. on Software Engineering, Nov. 1998.

[20] A. van Lamsweerde and L. Willemet, "Inferring Declarative Req. Spec. from Operational Scenarios," IEEE Trans. on Soft. Eng., Dec. 1998, pp. 1089-1114.

[21] E. Letier, et al., "Agent-Based Tactics for Goal-Oriented Req. Elaboration," in Proc. of 24th ICSE, 2002.

[22] K. Jain, et al., "Agents for Process Coherence in Virtual Enterprises," CACM, vol. 42, no. 3, March 1999, pp. 62-69.

[23] K. Jain, et al., "Using Spheres of Commit-ment to Support Virtual Enterprises," in Proc. of 4th ISPE Int'l Conf. on Concurrent Eng. Research and Applications (CE), Aug. 1997, pp. 469-476.