Decision Support System Based on High Level Architecture

A. Karkanitsa
Yanka Kupala State University of Grodno, 22 Ozheshko str. 230023 Grodno, Belarus,
karkanica@gmail.com

Abstract: *The paper describes the class of operational decision-making problems on base of distributed innovative knowledge. The purpose of research is to develop a method of constructing a dynamic subject domain. Unacceptability of use of static models of subject domain is proved. The possibility to automate the process of subject domain construction for this class of problems has been investigated. The model of a dynamic scene and methodology of its construction are proposed. The described methodology is based on the concepts of High Level Architecture (HLA) standard for distributed simulation systems and has been implemented by means of HLA Development Kit Framework.*

Keywords: decision making, dynamic subject domain, distributed innovative knowledge, High Level Architecture.

1. INTRODUCTION

The ever-changing business environment in today's global marketplaces, processes of formation of global information and communicative society, acceleration of business processes have led to emergence of the complex operational challenges requiring the fast solution and highly skilled expert knowledge. For example, complex products are often developed as a result of collaboration between many partners, each of which possesses expertise related to a specific industrial or research sector. Knowledge for the solution of such problems often is absent because of originality of each new problem. However, as a rule, they are available from experts of the research centers placed in the different countries of the world. The solution of a task is possible due to its decomposition on subtasks, distribution of subtasks between experts and the subsequent combination of the received results.

Examples of the modern operational (real-time) tasks are development of software products which are initiated in one country and realized in other countries by groups of performers; international scientific projects in which scientists from the different countries of the world are involved; management of the distributed organizations and others. Collaborative development has become a strategic issue to solve complex tasks at low cost and with quick response times to user demand. Critical parameter is time of the solution of a task which excess lead to receive an irrelevant result. Thus, there is a new class of tasks which can be defined as the problem of operational decision-making (PODM) on the basis of distributed innovative expert knowledge [1, 2].

At present, under the influence of globalization the properties of operational tasks have significantly changed. They assume the increasing scale, demand to involve a large number of experts. Efficiency of decision-making directly depends on as far as it will be possible to minimize time of knowledge receiving and processing. Quality of the decision depends on completeness and relevance of the expert's knowledge. The subject domain of operational tasks usually isn't defined at the initial stage of life-cycle. A subject domain model, number of subtasks and the involved experts are completely unknown [3]. Subject domain is not static any more.

Therefore we face a problem of development of methods and technologies of construction of subject domain for the modern decision-making problems. In this context, the paper presents High Level Architecture based approach that allows automating the process of subject domain construction of PODM. The key novel aspect of this approach is to build a dynamic scene of decision making problem as a Federation that includes a group of distributed Federates (HLA, стандарт IEEE 1516-2010) [4].

2. CASE STUDY

A subject domain is the cornerstone of the solution of any task. The life-cycle of the PODM includes following processes: 1) construction of the subject domain model; 2) model decomposition; 3) acquisition of expert knowledge; 4) formation and publishing of the subject domain.

Processes 1) - 4) are carried out within some scene. The scene is an environment that includes the participants (center, experts) and communications for exchange of information between participants. Communications infrastructure includes local networks and the Internet. The center and the experts are given an authority for further decomposition of an initial complex task into subtasks. The quantity of participants of the scene can be changed depending on results of main task decomposition and need of involvement of new experts.

In general, the class of tasks stated above has some specific properties:
1) variable number of subtasks to be solved;
2) variable number of experts to be involved;
3) partly known subject domain;
4) dynamic process of task solution.

Thus, we deal with dynamic subject domain, both from the point of view of its structure, and from the composition point of view (relevance and completeness of expert knowledge). In the theory of artificial intelligence the subject domain is named dynamic if the basic data describing subject domain changes during the solution of the task.

The review of functional, structural, object-oriented and other methodologies of subject domain construction allows drawing a conclusion that their application is inefficient in the conditions of 1) – 4). They are focused on well-structured tasks with the static model of subject domain, local databases and few participants of a scene. The algorithmic and technological components providing dynamic adaptation of the subject domain model are missing. Besides, use of the static subject domain models for operational problems of decision-making becomes unacceptable as it doesn't guarantee relevance and completeness of expert knowledge.

3. PROBLEM STATEMENT

Let's say there is some complex problem *Task* that includes the problem definition of the main problem S_0 and n of its atomic subtasks:

$$Task = (S_0, S^1, S^2, \dots, S^n),\ n \to \infty.$$

Subtask S^i is a set of four constituents: problem statement (*Text*), requirements to the solution (Spec), solution time limit (*Time*) and limit of material inputs (*Price*).

$$S^i = (Text^i, Spec^i, Goal^i, Time^i, Price^i).$$

Problem *Task* is operative, thus: $Time^i \to 0$, $Price^i \to 0$. As any modern problem *Task* is structured, decomposed into subtasks and can be solved by the (*Group)* of distant executors. *Group* consists of the center (*Center*), initiating the problem and distant experts (*E*) solving the problem.

Scene is a scheme including group of participants (*Center, E*), their roles and relations.

$$Scene = (S_0, Center, E^1, E^2, \dots, E^k),\ \ k \le n.$$

The *Center* formulates problem S_0, decomposes it into subtasks $S^1, S^2, \dots, S^n$, distant experts $E^1, E^2, \dots, E^k$ solve them in accordance with requirements *Spec*. Experts are known to the *Center* and competent. One-to-one correspondence between subtasks and experts is established. Z^i is a knowledge pattern necessary and sufficient for the solution of every subtask S^i. Set of knowledge patterns Z^i in the sum is the subject domain of *Task*. As it has been noted in the section 2, the subject domain is dynamic.

It is required to develop technology of construction of dynamic subject domain. A technology should be designed for automation of the following stages: 1) construction of subject domain model; 2) construction of the dynamic scene of the solution, having defined its participants, their roles and communications; 3) integration of a scene into the global infrastructure providing communications between participants; 4) development of algorithm for integrating a set of knowledge patterns received from the distributed experts into subject domain.

4. CONSRTRUCTION OF A SUBJECT DOMAIN

To solve the problem of construction of the subject domain model we propose a dynamic graph-model represented in the form of the connected attributed tree [5]. The model is constructed according to the principle of hierarchical decomposition of a target task for what we use one-to-one mapping between subtasks and tree nodes (Fig.1).

The unique identifier and set of attributes is assigned to each node *V*.:

$$V = \langle id, Attr^{id} \rangle,\ Attr^{id} = \langle task, name, addr, state, inf \rangle,$$

with *id* – unique identifier of subtask; *task* – a problem definition; *state* – subtask status (0 – the task is initiated, but not solved; 1 – the task in the course of the solution, 2 – the task is solved); *name* – the unique identifier of the expert; *addr* – the expert's address; *inf* – knowledge pattern, actually the solution of the task.

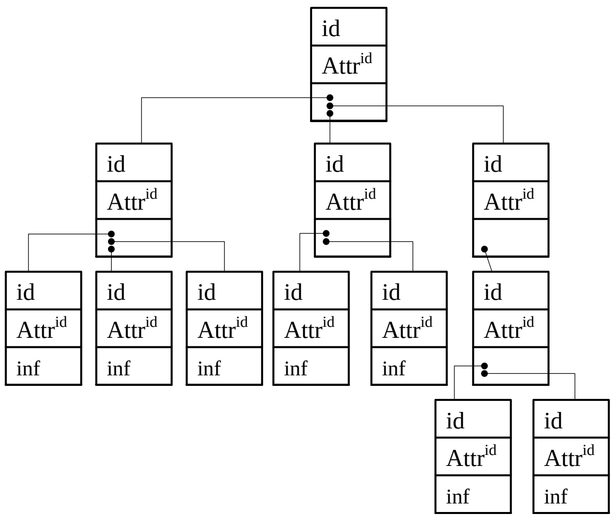

Fig.1 – Connected attributed tree model of SD

To solve the problem of construction of the dynamic scene it is necessary to meet the following set of requirements:

• unambiguous identification of participants of the scene;
• supporting the data exchange between the distributed participants by means of communications (local and global net);
• changing the quantity and the structure of scene participants;
• saving of hierarchical structure of subtasks for the subsequent correct integration of set of knowledge into the subject domain.

The method of construction of such a dynamic scene is proposed in this paper. The method is implemented on a basis of general purpose architecture for distributed computer simulation systems named High Level Architecture (HLA). The HLA technology has been chosen because of matching of task properties described above with the main concepts of HLA.

According to HLA, the simulated system should be presented in the form of federation. The federation is considered as set of all participants of simulation cooperating together to get the solution of some specific problem. Besides, the HLA standard provides interaction and a data exchange between the distributed components of system that meets the main requirement of a dynamic scene.

5. HLA BACKGROUND AND CONCEPTS

The High Level Architecture is the IEEE 1516-2010 standard, one of the most completed and popular standard for distributed simulation. Due to its capabilities to enable the interaction of distributed components, HLA is exploited in a great variety of application domains. For example, HLA-based software architecture specification can be applied to build complex environment to perform a collaborative research project.

In HLA standard, every participant is called *Federate*; the simulated system is called *Federation*, and it is composed of several HLA *Federates*. Federates interact using services proposed by the Run-Time Infrastructure (RTI). Several open-source and commercial RTI implementations are available today [6]. The RTI provides a set of services to manage the communications and data exchange among *Federates* conforming to the standard HLA specifications (Fig.2).

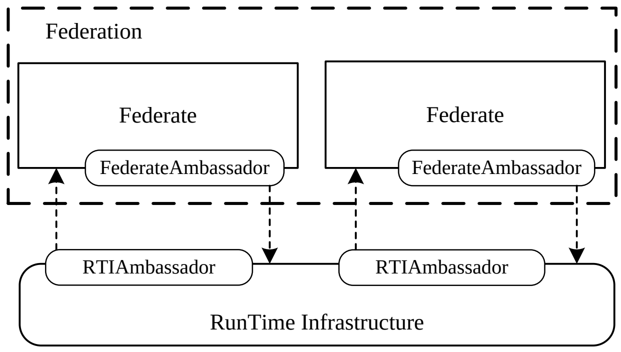

Fig.2 – Federate interaction through the RTI.

HLA provides two mechanisms for interaction and information exchange represented in a form of class: *Objects* and *Interactions*. An *object* is a collection of related data sent between federates, *interaction* is an event sent between simulation entities. The interactions between federates involve *objects* and *interactions* which work in a "Publish/Subscribe" model. A federate can register an object, which is instance of a class, and then change the values of the attributes of the object. Other federates that are subscribed to the class can discover the object and then receive attribute value updates.

HLA specification and RTI provide services that allow changing the structure of the federation dynamically during the federation execution. It is provided by infrastructure of RTI services - Join Federation Execution and Resign Federation Execution. This fact is essential to achieve the objectives of presented research. In terms of the problem definition stated in Section 3 of the paper it means that it is possible to involve an unlimited number of distributed experts in case when the task decomposition is required. Or we can resign an expert in case when a subtask is solved and the corresponding knowledge pattern is received.

The next section of the paper proposes a dynamic scene of the solution of the problem of subject domain construction on the basis of HLA concepts and RTI services.

6. CONSTRUCTION OF A DYNAMIC SCENE

Let's describe the general scheme of the solution of the *Task*, having included all participants of simulation. According to a problem definition the main participants are the *Center* and the group of local and distributed experts *E*. The *Center* sets a problem *S* for experts and expects a knowledge pattern. The subject domain is the solution of a problem *S* formed of the set of the received patterns. Communications between the center and experts are established as local or Internet-communications.

According to HLA concepts, we present the scene stated above in terms of HLA federation. The federation is formed of all participants (Center and Experts). Every scene participant is presented as an independent federate. Because the interactions between federates can be established only via the HLA RTI, we replace local and Internet-communications between participants of a scene with the RTI provided services.

At the same time the current IEEE 1516 HLA does not support any specification to describe a hierarchical structure of the federation. Federates are positioned as a part of federation but independent participants. There is no concept of federate autonomy. Therefore, if we simply follow the principle of federation execution and create a new federate for every new subtask and expert, we won't be able to construct the subject domain model corresponding to hierarchical decomposition of the main task. It means that when every federate solves the subtask and creates a pattern of knowledge of distributed expert, integration of these knowledge fragments into subject domain according to its model will be impossible.

Without breaking the general paradigm of HLA we offer to modify federate model by adding on more attribute - ownership attribute. If expert E_i performs decomposition of a subtask S_i, ownership attribute for federate E_i is represented as a list of unique indexes of subtasks S_i: $i_1, i_2, \dots i_n$. The same indexes are assigned to the corresponding attribute of a new created federates (Fig.3).

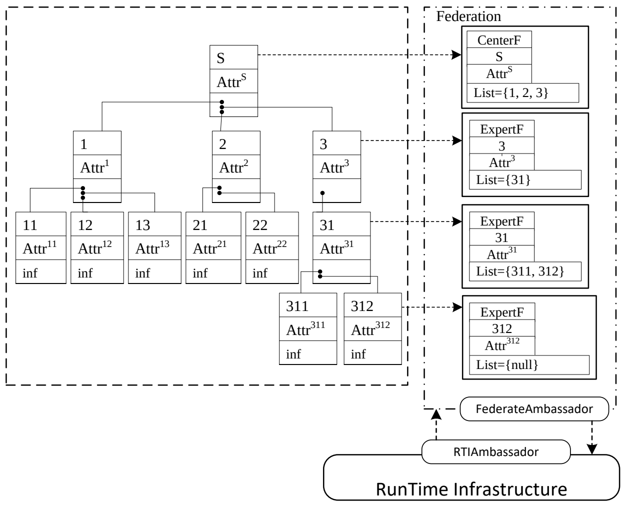

Fig.3 – HLA-based representation of the dynamic scene.

Dynamism of the scene and the corresponding modifications of the subject domain model are provided with Join Federation Execution and Resign Federation Execution services. Join-service creates a new federate and joins it to federation execution. Resign-service disconnects a federate.

Thus, the implementation of the scene on base of HLA architecture actually provides the mechanism of management of dynamic process of subject domain construction. The modified model of a federate guarantees a possibility to develop an algorithm of integration of distributed expert knowledge according to treelike model of subject domain.

7. FEDERATION LIFE-CYCLE

The developed models and scene allow describing a federation execution life-cycle. An ultimate goal of federation is a construction of subject domain from fragments of distributed expert knowledge. The process of subject domain construction is dynamic, the federation manages this process.

There are two types of federates in our scene: *CenterFederate* and *ExpertFederate*. Every *ExpertFederate* publishes some attributes, *CenterFederate* subscribes to attributes. During the federation execution *CenterFederate* observes the group of *ExpertFederate* and expects to get a fragment of knowledge.

The published attribute of the ExpertFederate is a autonomy coefficient *K* and can be equal to one of possible values (0, 0.5, 1). Autonomy coefficient of *FederateCenter* is equal to 0 by default. It means that the *CenterFederate* doesn't have enough resources to solve a target task and task decomposition will be executed, so new objects of type *ExpertFederate* are created and join Federation Execution. The *K*=0.5 for ExpertFederate has the same meaning. If *K*=1 then additional resources are not required to solve a subtask, task is solved and knowledge pattern is published, ExpertFederate mission is completed.

Thus, federation life-cycle can be presented as a dynamic process of joining of federates and observation of their autonomy coefficient value. At the moment when there is no any federate with *K*=0.5, the *CenterFederate* changes its own *K* to be equal to 1 and process of knowledge integration begins. During the integration the hierarchical structure of subject domain manages to be restored via federates ownership attribute.

8. FEDERATION DEVELOPMENT METHOD

The next problem which we had faced is that the development of HLA Federates are generally difficult, complex, and resource-intensive not only because of the complexity of the IEEE 1516-2010 family standards but also due to the lack of proper documentations and ready-to-use examples [7].

Thus, it would be desirable to use a software framework that simplifies the development of HLA Federates, manages the common HLA functionalities and provides high level functionality both to implement HLA Federates and manage the interactions between them and the RTI.

To simulate the scene of construction of dynamic subject domain we use the HLA Development Kit Framework (DKF) released under the open source policy [7].The DKF is implemented in the Java language and is fully compliant with the IEEE 1516-2010 specifications.

The architecture of a DKF-based Federation is composed of three main layers (Fig.4): (1) Application Layer, which contains the Federates that can interact with both the DKF and the HLA RTI by using their APIs; (2) DKF Layer, which represents the core of the architecture and provides a set of domain-independent; and (3) HLA RTI Infrastructure, which represents the RTI that host the Federation.

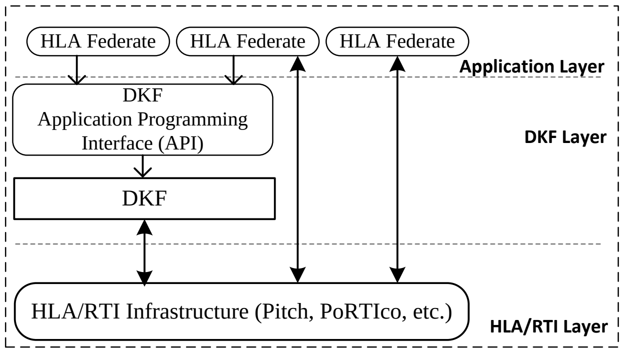

Fig.4 – The architecture of a DKF-based Federation

The process to build a Federate from scratch is composed by the following five main steps.

1. Build a model of the Federate that specifies the objects that the Federate manages, the attributes of these objects and the coders to handle such attributes.
2. Build a concrete Federate that specifies the behavior of the model defined at step 1. It is required to extend the AbstractFederate abstract class provided by DKF.

```
public class ExpertFederate extends SEEAbstractFederate
{
     private Expert expert = null;
     private String local_settings_designator = null;
     public ExpertFederate(SEEAbstractFederateAmbassador
seefedamb, Expert expert) {
       super(seefedamb);
       this.expert  = expert;}
}
```

3. Implement three methods according to the Federate life-cycle: configureAndStart() method; doAction() method that specifies the behavior of the Federate; an update method that specifies how to handle the RTI callbacks about the interactions that the Federate has subscribed.

```
public void configureAndStart(Configuration config) {
     // 1. Connect on RTI
     super.connectOnRTI(local_settings_designator);
     // 2. The Federate joins into the Federation
     super.joinIntoFederationExecution();
     // 3. Subscribe the Subject
     super.subscribeSubject(this);
     // 4. publish expert object on RTI
     super.publishElement(expert);
     // 5. Execution-loop
     super.startExecution();}

private void stopExecution() {
       super.unsubscribeSubject(this);
       super.diconnectFromRTI();}
@Override
protected void doAction() {}
@Override
public void update(Observable arg0, Object arg1) {}
```

4. Implement the Federate Ambassador extending the AbstractFederateAmbassador; typically, since no specific implementation is required, the child class has only to define its constructor which in turn calls the parent one.

```
public class ExpertFederateAmbassador extends
SEEAbstractFederateAmbassador {
       public ExpertFederateAmbassador() { super(); }
}
```

5. Implement a main class so as to instantiate and run the developed Federate.

```
public class Main {
     private static final File confFile = new File("conf.json");
     public static void main(String[] args) {
     Expert expert = new Expert("ExpertName","ExpertAddr");
     ExpertFederateAmbassador ambassador = new
     ExpertFederateAmbassador();
     ExpertFederate federate = new ExpertFederate(ambassador,
     expert);
     federate.configureAndStart(confFile);}
}
```

Usage of the described methodology allows creating a dynamic scene for the subject domain construction on the basis of distributed expert knowledge. It is obvious that HLA Development Kit Framework considerably simplifies the process of federate development. That is not a challenging task anymore and doesn't require considerable development efforts. There is no need to involve a large group of expert engineers with knowledge and experience in distributed systems, simulation, middleware and software programming.

9. CONCLUSION

Thus, the new class of problems of operational decision-making on the basis of distributed innovative expert knowledge is considered. The conclusion that was drawn is that the subject domain of such class of tasks is dynamic.

It is shown that there is a need of automation of process of subject domain construction. As a result, the problem of creation of dynamic subject domain is formulated and stages of its solution are listed. The dynamic scene of the solution is presented in terms and according to the architecture of HLA federation. For adaptation of a scene to the model of subject domain, modification of model of HLA-federate is performed. It has been done via adding the ownership attribute and autonomy coefficient.

The algorithm of construction of subject domain was described as an algorithm of HLA federation execution. It is proposed to perform the programming implementation of the developed dynamic scene with the use of HLA Development Kit Framework which significantly simplifies process of development of a federate without involving high skilled software developers. The methodology of this approach is described.

10. REFERENCES

[1] G. Shakah, V. Krasnoproshin, A. Valvachev. Decision Making System for Operative Tasks. *Proceedings of the X International Conference "Pattern Recognition and Information Processing"*, Minsk, Belarus 19-21 May 2009, pp. 272-275.

[2] S. Ablameyko, V. Krasnoproshin, A. Valvachev. Distributed Cognitive Resources as a Basis for Solving Operational Management Problems. *Proceedings of the XIX International Conference AEDEM 2010 "Global Financial & Business Networks & Information Management Systems"*, Minsk, Belarus 2010, pp. 337-342.

[3] V. Krasnoproshin, A. Valvachev. Technology for operative management of distributed organizations, Vestnik BSU. Series 1: Physics. Mathematics. Informatics, 1(1) (2009), p. 90-97

[4] IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA): 1516-2010 (Framework and Rules); 1516.1-2010 (Federate Interface Specification); 1516.2-2010 (Object Model Template (OMT) Specification).

[5] A. Karkanitca. Development of Dynamic Subject Domain Based on Distributed Expert Knowledge. *Proceedings of the Intern. Conference "Modeling and Simulation : MS'2012"*, Minsk, Belarus 2012, pp. 123-127.

[6] Pitch Technologies AB, "Pitch pRTI," 2016. [Online]. Available: http://www.pitch.se/products/prti

[7] A. Falcone. Easing the Development of HLA Federates: The HLA Development Kit and Its Exploitation in the SEE Project. *Proceedings of the 19th International Symposium on Distributed Simulation and Real Time Applications (DS-RT),* Chengdu, China 14-16 October 2015, pp. 50-57

[8] S.J.E. Taylor, P. Fishwick, R. Fujimoto, Panel on Modeling & Simulation Grand Challenges. *Proceedings of the the 2012 Winter Simulation Conference (WSC),* New York, 2012, pp. 1-15.