# Transparent Modelling Of Objects' Evolution

Dimitrios Theotokis[1], Anya Sotiropoulou[1], Georgios Gyftodimos[2]

[1]Department of Computer Science and Technology, School of Science and Technology, University of Peloponnese, GR 22100 Tripolis, Grecce, Fax +30 2710 372160
{dtheo,anya}@uop.gr
[2]Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Panepistimiopolis, Ilissia, GR 15784 Athens, Greece, Fax +30 210 7275214
geogyf@di.uoa.gr

## ABSTRACT

Supporting behavioural changes and in particular unanticipated ones is essential for achieving the behavioural evolution of objects. Object evolution is an important feature for war game simulation systems. This is the case because a unit's behavioural landscape may be altered during the course of a simulation either by virtue of the simulation itself of by user intervention. For instance, an armoured vehicle unit may become a scout and later an artillery observation unit. In this article we present how objects' evolution can be modelled and realised transparently using a role-based system as part of an High Level Architecture/Run-Time Infrastructure (HLA/RTI) federation.

## KEYWORDS

Behavioural changes, Behavioural landscape, High Level Architecture, Object evolution, Role modelling

## 1 INTRODUCTION

Although, the object-oriented paradigm better models the real world, it is widely accepted that it does not provide the adequate infrastructure for modelling the evolution of real world objects [5,7,8,14]. The static nature of the inheritance hierarchy compounded by issues related to the common ancestors dilemma, code scattering and tangling as well as the invariability of the self-reference are the reasons for which the evolution of objects and classes alike is inhibited [12]. Viewing information systems and in particular war game simulation systems as "living systems" [10] one can identify that object evolution is an essential feature such systems should support. Object evolution becomes important - if not essential - in military simulation systems and in particular battlefield simulation. This is attributed to the need for the accommodation of change, which in many occasions is unanticipated. In what follows we address the issue of objects' behavioural evolution in terms of roles, which are considered as behavioural adjustments or modifications of objects' initial behavioural specification. A framework that supports role-based object evolution, namely ATOMA [13,14], is utilised on top of an HLA/RTI [1] federation in order to propagate behavioural changes throughout a federation.

The remaining of this article is organised as follows: The benefits obtained from the use of roles as a means of accommodating behavioural changes is presented in section 2. In section 3 a description of the ATOMA framework and run-time system is given focusing on the modelling and implementation of behavioural changes in terms of roles. A brief comparison with other role-based approaches is also given. In section 4 the HLA/RTI architecture is presented in brief. Next, in section 5, the proposed approach is presented providing a description of how behavioural changes are accommodated throughout an HLA/RTI federation. Finally, section 6 concludes the paper and addresses future research directions on behavioural evolution, within an HLA/RTI federation.

## 2 MODELLING BEHAVIOURAL CHANGES

According to Sowa [9] conceptual structures are natural types, “that relate to the essence of entities” and role types “that depend on an accidental relationship to some other entity”. For instance, in a battlefield simulation an armoured vehicle is a natural entity. Viewed as a scout unit or as an artillery observation and control centre can depend on the needs of the simulation and the availability of resources. As such, a scout unit or an artillery observation and control centre are roles that can be played by an armoured vehicle. Roles may be played when the accidental relationship becomes effective, as many times as it becomes so, and more importantly they can be played in association with each other. In other words, an armoured vehicle may be at the same time a scout and an artillery observation and control unit. In fact, a natural entity may play as many roles as are applicable in a given context. For example, if the need occurs the same armoured vehicle may also become an infantry transport unit. Role acquisition and abolishment may be the result of human intervention or the result of changes in the context that the entity is found. Such changes may occur either because the internal state of the object changes, for instance the hill-climbing ability of a particular unit may be reduced when the unit receives an enemy fire hit, or because an object is viewed by other objects from another perspective, e.g. a unit becomes a scout unit.

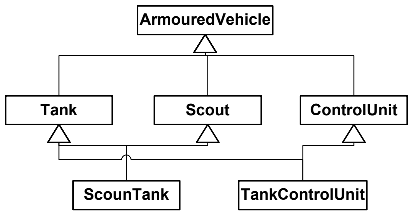

**Figure 1: An example**

In traditional object-oriented implementations and models all different behavioural scenarios must be explicitly modelled in terms of “kind-of” and “part-of” hierarchies or graphs. For instance, given the class hierarchy illustrated in Figure 1, if a tank unit is required to behave as a scout unit or as an artillery control unit, the corresponding subclasses must be *a priori* available. This imposes several problems:

1. All possible combinations of behavioural landscapes must be available during the design phase.
2. The resulting design introduces entities that do not really exist in the real world *per se*, but are only present to assist in resolving possible functional requirements
3. Class explosion results as the different behavioural landscapes played by natural entities (real-world entities) increases.
4. The resulting classes contain extrinsic behavioural characteristics thus introducing both code scattering and tangling.
5. There is no provision to accommodate unanticipated behavioural changes, except for redesigning the entire hierarchy to provide the necessary amendments.
6. For each particular behavioural requirement an instance must be created resulting in object schizophrenia, since the same unit is represented by two or more instances of different classes.
7. Deletion related problems: two or more instances of the same real world entity may co-exist, for instance a particular tank may exist on its own accord, but may also exist to model a scout vehicle. Apart from violating the underlying semantics related to instance existence, instance deletion (destruction) becomes inconsistent. When removing the instance of the tank-scout vehicle the tank part of it still remains as an instance of the tank class part. Moreover, if the instance of the tank is removed, due say to its destruction by enemy fire, the tank part of the tank-scout instance still remains.

The issue that causes the aforementioned problems is related to the lack of concern separation. It becomes evident that acquirable behaviour is time constraint and does not identify an object by virtue. It is only a means to an end, not the end itself as it accommodates for a temporary need. It is thus, and according to Sowa [5], a role type. Roles exist independently of the natural types (classes) and augment their behaviour when the need occurs and only then.

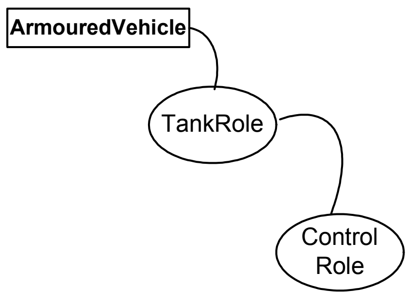

**Figure 2: Behavioural evolution**

Figure 2 illustrates the behavioural evolution of the problem area shown in Figure 1, in terms of roles. Class ArmouredVehicle represents the generic class for all armoured vehicles, whether tanks, scout units or artillery control units. The notions of a tank and artillery control unit are modelled as roles, which are assigned to instances of the class ArmouredVehicle, according to the requirements that exists at a given time.

To this extent the notion of roles employed here does not differ from those proposed by Gottlob et al. [4], Steinmann [10], Albano et al. [2], Bachman [3], Kristensen [6], amongst others. However, the realisation of roles and their underlying semantics present a number of issues addressed in the following section.

## 3 ROLES IN THE ATOMA FRAMEWORK AND RUN TIME SYSTEM

Under the ATOMA framework and run-time system classes and roles are two separate first class objects. Classes can exist on their own, while role instances exist in conjunction with the class instances they behaviourally modify. Role hierarchies are not supported by the framework at least at the definition layer. Role specialization is achieved by considering a role as a behavioural modification of an existing role. Key to the framework and run-time system is a three-layered architecture. The description layer provides the definitions and descriptions of classes and roles. The composition layer employs the mechanisms that enable the behavioural modification of classes in terms of roles, while the provision layer contains instances of both classes and roles.

What follows is a list of the key characteristics of roles and role playing under the ATOMA framework and run-time system:

1. A role comes with its own attributes and behavioural definition.
2. Roles depend on relationships with classes or class instances.
3. An object may play different roles at the same time.
4. An object may play the same role several times.
5. A role may be assigned to or removed from an object dynamically.
6. The removal of a role implies that all roles that may have been added to this role are also removed.

From an implementation perspective the behaviour of a role is not weaved into the object it behaviourally modifies. Instead dynamic proxies are constructed in order to achieve the behavioural modifications of an object by a role.

Role addition and activation are two separate procedures. A role may be added to an object but may remain inactive until it is explicitly activated. Role activation can occur either because an event triggering the role occurs, or because a condition holds or due to a combination of both. In this light role handling in the ATOMA framework differs from those in [4, 10].

## 4. HLA/RTI

The High Level Architecture provides an architecture for modeling and simulation. The intent of this architecture is to foster the interoperation of simulations and the re-use of simulation components. HLA is defined by three concepts:

- The object model templates
- The Runtime Infrastructure (RTI)
- The HLA compliance rules

The RTI and compliance rules are unchanged across all HLA-compliant simulations. However, each group of interacting simulations, or federation, must define a basis for the exchange of data and event between simulations.

## 4.1 Object Model Templates

The format and content of this basis is defined by the Object Model Templates (OMT). The definition of the Federation Object Model (FOM) is one part of the process of creating an HLA compliant federation. The Object Model Templates are used to describe the objects that will exist in the federation, the object attributes (the data that describe the state of the object), and the interaction that may occur between the objects in the federation.

## 4.2 The RTI

The RTI is a collection of software that provides commonly required services to simulation systems. These services fall into five categories:

- Federation Management
- Declaration Management
- Object Management
- Ownership Management
- Time Management

For the interested reader a more detailed description of HLA/RTI can be found in [1].

### 4.2.1 Definition of RTI federation

An HLA/RTI federation is the combination of a particular FOM, a particular set of federates and, and the RTI services. A federation is designed for a specific purpose using a commonly understood federation object model and a set of federates that may associate their individual semantics with that object model [1].

## 5. REALISING BEHAVIOURAL CHANGES IN A FEDERATE BASED SIMULATION

In order to accommodate for the behavioural evolution of objects participating in a distributed federate based simulation we have employed a role-based system, namely ATOMA [14], as a intermediate layer between each federate and its HLA/RTI component. Each federate's behavioural landscape is controlled by the role-based system. Role addition and removal occurs at this layer and is seemingly propagated to each federate via the HLA/RTI component. Transportation across the federation is achieved, as streams of byte-codes, which are resolved by each federate's ATOMA framework component. Figure 3 depicts schematically the architecture. Upon receipt of a behavioural modification the HLA/RTI component forwards the modification to the ATOMA layer, which is responsible for applying the behavioural change to the corresponding object in the federate leyer, for which it holds a representation.

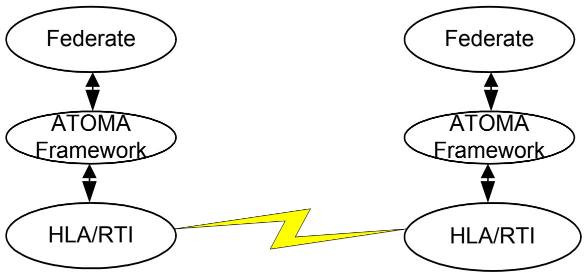

**Figure 3. System Architecture**

For example assume that within federate A an instance of a tank is assigned the role of a scout unit according to the semantics governing the Atoma run-time system described in section 3. This change is propagated to all the federates participating in the federation. Upon receipt of the appropriate message/event each federate's HLA/RTI component forwards this change to the federate's ATOMA layer, which applies the behavioural change to the appropriate object.

In order to achieve this functionality the Federation Object Mode (FOM) provides the necessary description for a behavioural modification. This description consists of an object whose attributes contain the following information:

1. The object that will be behaviourally modified. This is actually the object cross-federation unique identity.
2. The actual code that realises the behavioural modification. This is an array of bytes that is resolved to a role by the ATOMA layer, which is also responsible for applying the change to the corresponding object in each federate.
3. An indication whether the role should be activated immediately or not.
4. The condition(s) and event that will activate the role of a later time. This information is optional.

In terms of the FOM definition as depicted by a federation FED tile this information is structured as follows:

```
(class BehaviouralChange
      (attribute Recipient)
      (attribute RoleName)
      (attribute Code)
      (attribute Activation)
      (attribute ActivationRole)
)
```

When a role is to be activated at a later time this is achieved in terms of a message transmitted via the interactive mechanism supported by the RTI infrastructure. The message has the following structure:

```
(class RoleActivation reliable
      (parameter RoleName)
      (parameter Recipient)
)
```

## 6. CONCLUSIONS

The use of role-based systems for modelling and realizing the evolution of objects' behavioural landscape provides a natural and intuitive approach for realising "living" information systems. Role carry the semantics of a changing world and in doing so aid in the modelling of the system that abide to change, particularly when considering the time dimension. Roles under the ATOMA framework differ from other proposed role-based systems in two ways: The first one, although an implementation related one, bares some important semantic issues. An object does not directly relate to the roles it plugs at a given time. It only relates to a proxy for all its roles. Consequently, it is only responsible for delegating behaviour related requests, that it cannot handle, to its proxy, which in turn directs them to the corresponding roles. This provides a multi-role invocation scheme a characteristic present in real world "living systems". The second on concerns the activation semantics for roles. Existing role systems handle role activation at the same time of role acquisition and role deactivation at the time of role removal. Under the ATOMA run-time system these two notions are separate, thus providing a degree of flexibility missing in most role-based systems. Coupled with HLA/RTI such a system supports the behavioural evolution of a federate-based military simulation system, thus providing a more realistic representation of the simulated objects.

## REFERENCES

[1] DMSO https://sdc.dmso.mil/

[2]A.Albano, R.Bergamini, G.Ghelli, & R.Orsini, An object data model with roles, in: R.Agrawal, S.Baker, D.Bell (Eds) Proceedings of the 19th International Conference on VLDB, Morgan Kaufmann, Dublin; 1993 pp.39-51

[3]C.W.Bachman & M.Daya, The role concept in data models, in Proceedings of the 3$^{rd}$ International Conference on VLDB, 1977;pp.464-476

[4]G.Gottlob, M.Schrefl & B.Röck, Extending object-oriented systems with roles, ACM TOIS 14(3) (1996) pp. 268-296

[5] G.Kiczales. E.Hilsdale, J.Hugunin, M.Kersten, J.Palm &W.Griswold. (2001) An overview of AspectJ. In Proc. Of 15th. ECOOP, LNCS 2072, p. 327-353, Springer-Verlag,

[6] B.B.Kristensen, Object-oriented modelling with roles in J.Murphy, B.Stone (Eds) Proceedings of OOIS 95, 18-20 December 1995, Dublin, Springer 1996, pp.57-71

[7] M.Mezini (1998) Variational Object-Oriented Programming Beyond Classes and Inheritance, Kluwer Academic Publishers

[8] H.Ossher & P.Tarr. (2000) Multi-Dimensional Separation of Concerns and the Hyperspace Approach. In Proceedings of the Symposium on Software Architectures and Component Technology: The State of the Art in Software Development. Kluwer.

[9] J.F.Sowa. (1984) Conceptual Structures: Information Processing in Mind and Machine, Addison-Wesley, New York

[10]D.Stamoulis, D.Theotokis, D.Martakos, & G.Gyftodimos. Ateleological Development of Design Decisions Independent Information Systems. In Adaptive Evolutionary Information Systems, Editor: Nandish Patel, IDEA Publishing Group, ISBN 1-59140-034-1.

[11]F.Steinmann On the representation of roles in object-oriented and conceptual modelling, Data & Knowledge Engineering 35(2000) 83-106

[12]A.Taivalsaari (1996). On the notion of Inheritance, ACM Computing Surveys, Vol 28 Number 3, pp 438-479

[13]D.Theotokis. G.D.Kapos, C.Vassilakis, A.Sotiropoulou & G.Gyftodimos, Distributed Information Systems Tailorability: A Component Approach in Proceedings of the IEEE Workshop on Future Trends on Distributed Computing, Cape Town, 1999, pp. 95-101

[14]D.Theotokis. (2003) Approaching Tailorability in Object-Oriented Information Systems through Behavioural Evolution and Behavioural Landscape Adaptability to appear in the Journal of Applied Systems Studies special issue on "Living Evolutionary and Tailorable Information Systems: Development Issues and Advanced Applications