Roles:From Objects, Security and Databases Perspective

Chandra Sekharaiah Kangaluru
Distributed Object Systems Group
Department of Computer Science & Engg.
JNTU College of Engineering Hyderabad, India
shakes123@sify.com

D.Janaki Ram
Distributed Object Systems Lab
Department of Computer Science & Engg.
Indian Institute of Technology Madras
d.janakiram@cs.iitm.ernet.in

Abstract

A role paradigm conformance model (RPCM) called Typehole model was developed and implemented in Java. It does not use Is-role-of inheritance and is free from the object schizophrenia (OS) problem (OSP). An extended distributed role model provides multifarious role hierarchies at different sites. Modeling object role systems is prone to OSP. OSP is the error in the RPCM design to subscribe to the viewpoints in the client space. OS and OSP are contrasted. A classification of OSPs is made. Role modeling problem is the requirement to design an OSP-free RPCM with Is-role-of inheritance. The existing object-oriented role models have been evaluated as regards OSP. No role model has strong claims as regards solving the role modeling problem. A security model was developed for RPCM-based distributed object systems and implemented on CORBA. By designing message filter hierarchies, hybrid security is developed at global and local levels for RPCM-based object systems. Role is either abstract or concrete as it can be a type or an instance.

Roles: The Diverse Approaches

Roles community have taken diverse approaches in providing the role mechanism support. In this section, we present the definitional details of the role mechanism, roles properties and the approaches taken by the pattern community and the database community in extending objects with roles.

The object community in general agree on the significance of the concept of role for object modeling. Nevertheless, there is little agreement about the definition and semantics of role; for example, the standardization experts of RM-ODP are not sure of whether role is a type or an instance concept. This is found an issue of confusion in the UML standard, and prevents ISO experts from a broad consensus and finalizing a language for ODP enterprise modeling. Roles are also of consideration by cognitive experts (Chandra Sekharaiah K. & Upakaram Gopal 2004) for understanding psycho-neuro systems and in system modeling (Upakaram Gopal & Chandra Sekharaiah K. 2003).

Role: The Definition Problem

In (Guy Genilloud & Alain Wegman 2000a), the current RM-ODP definition of roles is:

Role: Identifier for a behavior, which may appear as a parameter in a template for a composite object, and which is associated with one of the component objects of the composite object.

Specification of a template as a composition of roles enables the instantiation process to be explained as the association of a specific component of the resultant composite object with each role. The association of a component object with a role may result from the actualization of a parameter.

Further, role is considered as an identifier for a behavior, as a named behavior and as a general tool for design. Role is also considered as a type concept and also as an instance concept. In (Li, Q. & R. K. Wong 1999), "In particular, an object at any time point is described not only by an instance of a class, but also an instance of every role type whose role it currently plays". In (Trygve Reenskaug 1995), Trygye Reenskaug, the inventor of role modeling for software engineering, uses the term role to denote an object type: "A role in an object specification is called an object type, which is a specification of a set of objects with identical, externally visible properties." According to him, roles have all the properties of objects, roles are not types, but can be typed. This indicates that there exist both a concept of role and a related concept of role type. It is observed that Reenskaug uses the same term "role" in different contexts to refer to an instance concept or to a type applicable to objects. Such a practice is found quite common and unavoidable. For example, the UML standard uses the term "actor", "action", "event" etc. to denote either an instance concept or an occurrence concept or a type concept. In (Grady Booch, James Rumbaugh, & Ivar Jacobson 2000), a class participating in an association is said to play specific role in that relationship; a role is just the face the class at the near end of the association presents to the class at the other end of the association. The role class in an association is given an explicit name. Further, the same class can play the same or multifarious roles in other associations. UML defines role as the named specific behavior of an entity participating in a particular context and, thus, to an instance concept. Thus, role is one term for several concepts.

In (Guy Genilloud & Alain Wegman 2000a; 2000b), the

poorly formulated RM-ODP definition of role was revised as:

Role: An abstraction of the behavior of an object that consists of a subset of the interactions of that object together with a set of constraints on which they may occur. A role always belongs to a specific larger behavior that involves other roles, called a *collaborative behavior*.

Understanding the Role Definition Further

In (P.Papazoglou & B.J.Kramer 1997), a role refers to the ability to adapt the classification of an object, so that the same object can be an instance of different classes some of which are created dynamically. A role is an interface-based specification implemented on the basis of pre-existing objects in a way that allows a pre-existing object to assume or relinquish behavior dynamically while retaining its original identity. In the object database models (P.Papazoglou & B.J.Kramer 1997), roles are used to facilitate the migration of objects in the class DAG. In such role models, the objective of roles is to customize objects to application needs so that they become equipped with their own idiosyncratic behavior. However, they are different from views in that their objective is to provide dynamic object migration and automatic re-classification without affecting the database schema.

Roles Properties

The new definition was checked against the properties of roles detailed in (Steimann 2000) as given below.

- Roles export role-specific behaviors.
- Roles depend on is-role-of relationships
- An object may play multifarious roles simultaneously
- An object may play the same role several times, simultaneously
- An object acquires and relinquishes roles dynamically
- The sequence in which roles may be acquired and abandoned can be subject to restrictions.
- Objects of unrelated types can play the same role.
- Roles can play roles
- The state of an object can be role-specific
- An object can export role-specific features
- Roles provide access control
- Different roles may share structure and behavior
- An object and its roles share identity
- An object and its roles have different identities
- A role can be transferred from one object to another

Role Paradigm Conformance Model: The Core Feature Set

Various important characteristics for objects with roles have been explored in the literature (George Gottlob, Michael Schrefl, & Brigitte Röck 1996; Bent Bruun Kristensen 1995; Tamai 1999). A core set of essential features has been identified for objects with roles. They are explained below. The feature set is called role paradigm. A role paradigm conformance model (RPCM) satisfies all the features in the set. An object with roles that conforms to the role paradigm is called an RPCM object. The characteristic features of an RPCM object are explained below.

- Per-role visibility of the object: A role exports the role-specific behavior of an object. Controlled visibility and access of the object should be possible on per role basis. That is, an object's methods can be accessed and manipulated in terms of a role and excluding those of other roles. This is also seen as the possibility of classification of an object with multiple, disjoint properties.
- Multifarious roles: An object may play roles of multiple role types.
- Dependency: Roles are invariably existence-dependent on the core object which is the role-playing object. The methods of the role may be defined in terms of the methods of the object, but not vice versa.
- Integral identity: An object and its roles have one identity. They are seen and manipulated as one and the same entity.
- Polyinstantiation of roles: An object may play multiple roles of the same type simultaneously.
- Dynamic evolution: Roles can be acquired and abandoned dynamically during the lifetime of an object.
- Abstractional hierarchy support: The entity and its roles can be organized in an abstraction hierarchy such as generalization and aggregation hierarchy.

In the realm of role modeling, object orientation has taken an evolutionary path towards the design and implementation of role paradigm conformance models. The work in (George Gottlob, Michael Schrefl, & Brigitte Röck 1996) details a well-known RPCM implemented in Smalltalk extended to support roles. It combines the object inheritance model of the prototypical approach in the class-based object-oriented language. Role model synthesis was proposed in (Trygve Reenskaug 1995). In OOram methodology, a role model is a part of a structure of objects which is chosen to regard as a whole, separated from the rest of the structure during some period of consideration. A whole that is chosen to consider as a collection of roles, each role being characterized by attributes and by actions which may involve itself and other roles.

Objects with roles have to satisfy such properties as visibility, dependency, identity, multiplicity, dynamicity, and abstractivity (Bent Bruun Kristensen 1995; Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000; Chandra Sekharaiah K., Arun Kumar, & Janaki Ram D. 2002). The properties were also explained in (George Gottlob, Michael Schrefl, & Brigitte Röck 1996). These properties have been stated time and again in literature though descriptions are different. This set of properties is much akin to the role paradigm (Chandra Sekharaiah K., Arun Kumar, & Janaki Ram D. 2002). The works in (George Gottlob, Michael Schrefl, & Brigitte Röck 1996;

Bent Bruun Kristensen 1995) explain how traditional object oriented techniques such as specialization, aggregation and association are not adequate to capture all the properties in the role paradigm. Role paradigm conformance is a requirement for advanced role models. A role model which conforms to the role paradigm is called *role paradigm conformance model* (RPCM). The role models in (George Gottlob, Michael Schrefl, & Brigitte Röck 1996; Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000) are RPCMs.

The role models in (Joel Richardson & Peter Schwarz 1991; Barbara Pernici 1990) are early, rudimentary role models. They do not use role object identifiers and, hence, do not satisfy the multiplicity characteristic. Even some later role models (Daniel Bardou & Christophe Dony 1996; Albano A., Ghelli G., & Orsini R. 1995) do not satisfy the multiplicity characteristic. Consequently, they are not RPCMs. A comparison of the inadequacy of many role models regarding role paradigm conformance may be found in (G.Kappel, W.Retschitzegger, & W.Schwinger 1998). The works on modeling objects with roles show an evolutionary path of advancement toward RPCMs. A role-playing object that conforms to the role paradigm is called an RPCM object. To be clear, in this work, by role modeling, we mean modeling RPCM objects.

In Figure 1, the RPCM object, John, plays Faculty role, Student role, Conference Associate sub-role, Lab-in-charge sub-role and Reviewer sub-role. S, F, ECOOP, OOPSLA, L1, OSP_paper are role instances and sub-role instances. An RPCM object has intrinsic and extrinsic properties and state. The core object, John, captures the intrinsic properties. These properties are time-invariant during the object lifetime. The role and sub-role objects capture the extrinsic properties and state. They are time-variant during the lifetime of the core object. The lifeline of the core object is the lifeline of the role/sub-role objects.

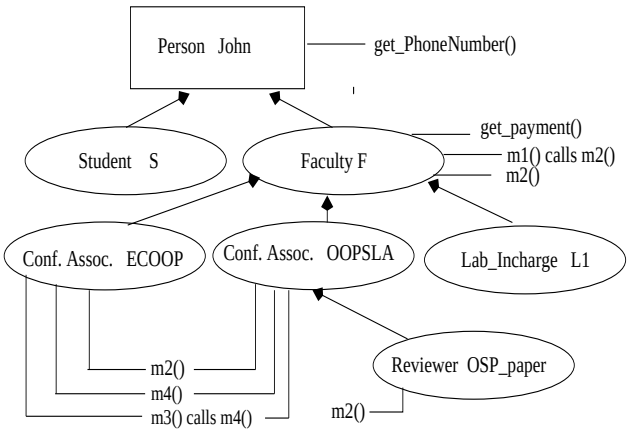

Figure 1: An Example Role Modeling Scenario: John Playing Roles

We find that the patterns community, the object-oriented role modeling community and the database community have taken different approaches as regards the role mechanism support for object-oriented software development.

Object-Oriented Modeling with Roles

Traditional objects are monolithic units. Objects can communicate with each other by playing certain roles for each other in these relations and interactions. In (Bent Bruun Kristensen 1995), roles of objects are a means to refine the understanding of an object as a monolithic unit. In the subsections, the patterns community approach and database community approach to roles are surveyed.

Roles: Patterns Community Approach

Roles are an effective mechanism to develop new patterns and to revise existing patterns. The works in (Peter Coad 1992; Erich Gamma *et al.* 1995; Martin F. Fowler 1997; Dirk Riehle 1997; Bent Bruun Kristensen & Johnny Olsson 1996; Erich Gamma 1996; Dirk Baumer, Wolf Siberki, & Martina Wulf 1997; Liping Zhao & Ted Foster 1999) show how roles can be related and used in some design patterns. In (Bent Bruun Kristensen & Johnny Olsson 1996), DECORATOR is considered the best design pattern that captures the properties of the role mechanism. Further, the design patterns, STATE, STRATEGY and BRIDGE, can be restructured into role-based structures with various benefits. Thus, the role mechanism may introduce new patterns. Roles are a means for composition-decomposition of concerns. The patterns in (Erich Gamma *et al.* 1995) can not solve the problems of separation of concerns adequately in OO modeling because the composability features of the patterns are defined by the capabilities of the conventional OO model (Mehmet Aksit 1996).

In an object-role hierarchy, role objects and sub-role objects are agents for the root object. The root object is the object which plays roles. Role-of relationship exists between the root object type and the types of the role objects. Similarly, role-of relationship exists between the types of role objects and the types of sub-role objects.

Object schizophrenia problem can occur in the design patterns of GOF (Erich Gamma *et al.* 1995) and some role-related patterns in (Peter Coad 1992; Dirk Riehle 1997; Erich Gamma 1996; Dirk Baumer, Wolf Siberki, & Martina Wulf 1997; Martin, R.,, D. Riehle, & F. Buschmann 1998) such as Active Bridge, Bureaucracy, Roles Played, Extension Objects, Role Object patterns. We give the results of our findings on various object schizophrenia problems (Chandra Sekharaiah K. & Janaki Ram D. 2002b; 2002c) in terms of classification of OSPs and degree of OSP. The role diagram for the Bureaucracy pattern is provided in Figure 2 with a notation extended from that in (Dirk Riehle 1997). The pattern features five roles- Director, Manager, Subordinate, Clerk and Client. Clients send messages to the lowest level role, Clerk. These entities perform the actual work or service for a client. The collaboration path between two roles is indicative of the processing of the delegation mechanism. Since the pattern employs message forwarding mechanism support, it is prone to OSP. In the Bureaucracy pattern, in effect, four of the GOF patterns govern the interaction of the objects. A Manager or Director who receives

a request, delegates work to its subordinates, as in the Composite pattern: the clerk plays the role of Component and the Manager the role of Composite. A Manager or Clerk receiving a request it can not handle forwards the request up the hierarchy, as in Chain of Responsibility: the Manager and Clerks play the Predecessor role and the Director the Successor role. Clerks or Managers who want to interact with each other first address their superior to coordinate them, as in the Mediator pattern: at the same hierarchy level the objects are Colleagues whereas the superior acts as Mediator. Finally, when a subordinate changes state, such as completing some work, or being absent, it reports this change of state to its superior. Thus, the superior acts as Observer of its subordinate Subjects, as in the Observer pattern. Roles describe the responsibilities of objects in a collaboration, but how a role is modeled or specified is often left open.

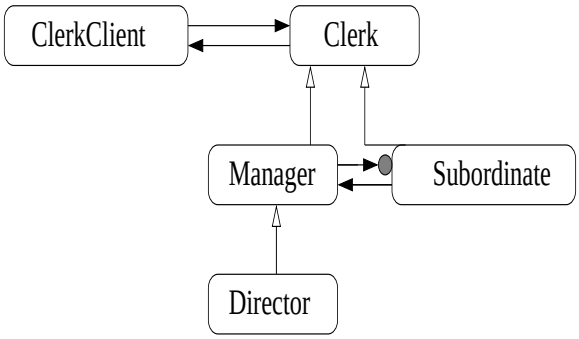

Figure 2: Role Diagram for the Bureaucracy pattern

Role mechanism support is provided mainly for three different purposes- for access control, to support object evolution by dynamic object composition-decomposition and for support of multiple views. The significance and support of multiple perspective/context-dependent behavior of objects were explained in (Joel Richardson & Peter Schwarz 1991; Barbara Pernici 1990; John J. Shilling & Peter F.Sweeney 1989). Classification of roles and encapsulation of roles as classes were not discussed in these approaches. Further, they do not adopt either inheritance or delegation between roles. Hence, role sharing among classes is not considered. There is no support of explicit operators for switching between roles. These role models are called early role models. They do not satisfy all the necessary properties for roles which were later identified in the literature.

The role models in (George Gottlob, Michael Schrefl, & Brigitte Röck 1996; Albano A. *et al.* 1993; Raymond K. Wong, H.Lewis Chau, & Frederik H.Lochovsky 1997) are progressive models in the evolutionary path of the role modeling literature toward role paradigm conformance models which satisfy all the necessary properties of roles. The role model in (George Gottlob, Michael Schrefl, & Brigitte Röck 1996) is significant in that it satisfies all the necessary properties for roles.

Roles: Database Community Approach

In the past three decades, several data models came up to capture the evolving and multifaceted nature of real world entities. The concept of role found place in computer science first in 1977 (C.W.Bachman & M.Daya 1977). It was aimed to enhance the expressive power of network data modeling. In this work, a role is a ”behavior pattern which may be assumed by entities of different kinds.” Playing roles of roles is not considered and have no concept of role- or object identifier. In the work in (C.W.Bachman & M.Daya 1977), the authors do not consider the possibility of roles being played by other roles.

The restriction that an object be associated with a single, most specific type in the database context was relaxed in Iris (Fishman, D.H. *et al.* 1987). The feature in it allows an object to belong to many types, but it misses the possibility of role-specific (context-dependent) behavior.

In the last decade, significance of roles was realized in work on office modeling (Barbara Pernici 1990), semantic modeling (Joel Richardson & Peter Schwarz 1991; Edward Sciore 1989), object-oriented modeling (George Gottlob, Michael Schrefl, & Brigitte Röck 1996; Bent Bruun Kristensen 1995) and multimedia applications (Raymond K. Wong 1999).

Typehole Model: An OSP-free Role Model

Roles are an effective mechanism to develop objects with multiple views, for dynamic object evolution and access control. We have investigated at modeling objects that play dynamically varying multifarious roles. Role-playing objects have to satisfy such necessary properties as per-role visibility, multifarious roles, dependency, integral identity, multiidentity, polyinstantiation, dynamicity, locality, and abstractional hierarchy support (Chandra Sekharaiah K., Arun Kumar, & Janaki Ram D. 2002; Chandra Sekharaiah K. & Janaki Ram D. 2002b; 2002a; 2002c; 1998; Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000). The importance of such necessary characteristics is emphasized in (George Gottlob, Michael Schrefl, & Brigitte Röck 1996). A role model that satisfies these properties is called Role Paradigm Conformance Model (RPCM) (Chandra Sekharaiah K. & Janaki Ram D. 1998; Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000). The existing work on role models shows an evolutionary path towards RPCMs. The earlier rudimentary role models did not use any message forwarding mechanism such as delegation. They did not use Is-Role-of inheritance (Chandra Sekharaiah K. & Janaki Ram D. 2002b). The later role models (Albano A. *et al.* 1993; Daniel Bardou & Christophe Dony 1996) used delegation. Neither kind of role models belong to the category of RPCMs. An object with roles that conforms to the role paradigm is called an RPCM object. Traditional object oriented techniques such as inheritance, aggregation and association are not adequate to model RPCM objects (Bent Bruun Kristensen 1995; George Gottlob, Michael Schrefl, & Brigitte Röck 1996). The role models in (Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000; George Gottlob, Michael Schrefl,

& Brigitte Röck 1996) are advanced role models in that they are RPCMs.

We have developed an RPCM called Typehole model (Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000). In this model, a class called root class abstracts two kinds of behaviors. First, it abstracts time-invariant, role-independent behavior called root behavior. Second, it abstracts time-variant, role-specific behaviors of the root object by providing Typeholes. Formally, a Typehole is a set of declarations for attributes and methods that are specific to a role or subrole (Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000). It is declared in root class to defer the definition of a role-specific or subrole-specific behavior. Typehole definition is given in a separate class called glue class. Different glue classes provide different type definitions for a role Typehole. Thus, the role-playing objects modeled after the Typehole model have the capability to take on different behaviors for any particular role. Role type can have different implementations (Albano A. *et al.* 1993). The Typehole concept facilitates such separation between role type and role implementation. Role interface and security filter interface (Chandra Sekharaiah K., D.Janaki Ram, & A.V.S.K.Kumar 2000) are captured as one and the same Typehole in the root class. Implementations are provided in the glue classes. Implementations in the glue classes can be dynamically bound to the role interfaces declared in the root class. This supports role polymorphism. In the Typehole model, messages are sent to particular, destination roles directly. It does not use Is-role-of inheritance and is free from the object schizophrenia problem (Chandra Sekharaiah K. & Janaki Ram D. 2002b; 2002a; 2002c). The model is used for dynamic composition and decomposition with roles and facilitates object evolution with roles. The model supports all the operational dynamics on roles such as create_role(), delete_role(), suspend_role() and resume_role(). It was implemented in Java. In a distributed role model, the multifarious roles of an object may be existent at different sites. In another approach, multifarious role hierarchies are existent at the different sites.

Object Schizophrenia Problem: Many Faces

We have investigated the problems related to the RPCMs. Object schizophrenia problem (OSP) (Chandra Sekharaiah K. & Janaki Ram D. 2002b; 2002a; 2002c; Upakaram Gopal & Chandra Sekharaiah K. 2003) has been examined. Modeling RPCM objects is prone to OSP. OS is an essential condition of an object characterized by broken semantics of identity, state, implementation, and contracts. OS by itself is not a problem. OSP is the inability of an object under OS to respond to the messages with proper information in spite of its availability. Such an inability is due to errors in RPCM design to subscribe to the viewpoints in the client space. We built a stronger notion of OSP. Basically, a sharp contrast was brought out between the notions of object schizophrenia and object schizophrenia problem. Different OSPs were classified as WMI-OSP, BD-OSP, BC-OSP, SS-OSP, DG-OSP, BA-OSP due to wrong method interpretation, broken delegation, broken consultation, security schizophrenia, dopplegangers and broken assumptions respectively (Chandra Sekharaiah K. & Janaki Ram D. 2002b; 2002c). Role modeling problem is the requirement to design an OSP-free RPCM with Is-Role-of inheritance (Chandra Sekharaiah K. & Janaki Ram D. 2002b). No existing role model has strong claims as regards solving the role modeling problem. OS is necessary and acceptable in RPCM designs and hence it is said, "Model OS, not OSP". We set this as the design goal for RPCMs in object oriented systems. A solution to OSP is found to be only model-specific. Eliminating OSP of one kind results in OSP of another kind. Hence, a complete solution to the role modeling problem is rather not possible. Degree of OSP is the measure of OSP complexity in a role model design. Our findings on the contrast between OS and OSP set a trend to the next generation role modeling community to address the role modeling problem thoroughly.

A Security Model for Distributed Object Role Systems

Modeling security for systems involving RPCM objects is not possible with the traditional object oriented approach. A security model was developed for RPCM-based object oriented systems (Chandra Sekharaiah K., Arun Kumar, & Janaki Ram D. 2002). By designing message filter (Rushikesh K. Joshi, N.Vivekananda, & D.Janaki Ram 1997) hierarchies similar to the concept of role hierarchy in the Typehole model, security is developed for RPCM objects at multiple levels. Message filters (Rushikesh K. Joshi, N.Vivekananda, & D.Janaki Ram 1997) are intermediate objects which intercept the messages on-the-fly to determine on the delivery of messages to objects. A message sent to a role object in the system is secured at global level and also at local level by message filters. The security model was implemented on CORBA. A distributed security model for an RPCM-based distributed object role system involves the modeling of multiple role hierarchies and the corresponding message filter hierarchies at different sites. This leads to very high degree of SS-OSP in the system.

Conclusions

The usage of message forwarding mechanisms in OS-prevailing object systems causes OSP. OS is inevitable in developing distributed object systems with roles. Typehole model is an OSP-free RPCM. It does not use Is-Role-of inheritance. There is no OSP-free RPCM that captures Is-Role-of inheritance. A solution to OSP is only model-specific. Eliminating OSP of one kind introduces OSP of another kind. Hence, a complete solution to OSP is rather not possible. Addressing security schizophrenia problem in RPCM-based object systems has good potential for research.

References

Albano A.; Bergamini B.; Ghelli G.; and Orsini R. 1993. An Object Data Model with Roles. In *Proceedings of the 18th VLDB Conference*, 39–51.

Albano A.; Ghelli G.; and Orsini R. 1995. Fibonacci : A Programming Language for Object Databases. *The VLDB Journal* 4(3):403–414.

Barbara Pernici. 1990. Objects with Roles. In *IEEE/ACM Conference on Office Information Systems ACM SIGOIS*, volume 1, 205–215.

Bent Bruun Kristensen, and Johnny Olsson. 1996. Roles and Patterns in Analysis, Design and Implementation. In *Proceedings of International Conference on Object Oriented Information Systems*, 242–263.

Bent Bruun Kristensen. 1995. Object Oriented Modeling with Roles. In *Proceedings of the Second International Conf. on Object Oriented Information Systems(OOIS)*, 57–71. Dublin, Ireland: Springer Verlag.

Chandra Sekharaiah K., and Janaki Ram D. 1998. Dynamically Mutable Multiple Abstractions. In *Proceedings of the Second National Conference on Object Oriented Technology (NCOOT'98)*, 26–32.

Chandra Sekharaiah K., and Janaki Ram D. 2002a. Modeling Roles and Security in RPCM-based Distribbuted Object Role Database Systems. In *PhDOOS 2002 Workshop, 16th European Conference on Object-Oriented Programming*.

Chandra Sekharaiah K., and Janaki Ram D. 2002b. Object Schizophrenia Problem in Modeling Is-Role-of Inheritance. In *Inheritance Workshop, 16th European Conference on Object-Oriented Programming*, 88–94.

Chandra Sekharaiah K., and Janaki Ram D. 2002c. Object Schizophrenia Problem in Object Role Database System Design. In *Object Oriented Information Systems (OOIS'02)*, 494–506.

Chandra Sekharaiah K., and Upakaram Gopal. 2004. Understanding Psycho-Neuro Systems from a Systemic Vs. Individualistic Perspective. In *International Conference on Cognitive Systems*.

Chandra Sekharaiah K.; Arun Kumar; and Janaki Ram D. 2002. A Security Model for Object Role Database Systems. In *International Conference on Information and Knowledge Engineering (IKE'02)*, 381–386.

Chandra Sekharaiah K.; D.Janaki Ram; and A.V.S.K.Kumar. 2000. Typehole Model for Objects with Roles in Object Oriented Systems. In *Fourteenth Europeon Conference on Object Oriented Programming (ECOOP 2000), LNCS 1964*, 301–302. Sophia Antipolis, France: Springer Verlag.

C.W.Bachman, and M.Daya. 1977. The Role Concept in Data Models. In *Proceedings of the 3rd International Conference on Very Large Databases*, 464–476.

Daniel Bardou, and Christophe Dony. 1996. Split Objects: A Disciplined Use of Delegation within Objects. In *Proceedings of OOPSLA*, 122–137.

Dirk Baumer; Wolf Siberki; and Martina Wulf. 1997. The Role Object Pattern. In *Pattern Languages of Programming Conference (PLOP'97) Technical Report 97-134*.

Dirk Riehle. 1997. Composite Design Patterns. In *Proceedings of OOPSLA*, 218–228.

Edward Sciore. 1989. Object Specialization. *ACM Transactions On Information Systems* 7(2):103–122.

Erich Gamma; Richard Helm; Ralph Johnson; and John Vlissides. 1995. *Design Patterns*. Addison-Wesley.

Erich Gamma. 1996. The Extension Objects Pattern. In *Pattern Languages of Programming Conference (PLOP'96)*.

Fishman, D.H.; H.P. Cate; E.C. Chow; T.Connors; J.W. Davis; N.Derrett; C.G. Hoch; W. Kent; P. Lyngbaek; B. Mahbod; M.A. Neimat; T.A. Ryan; and M.C. Shan. 1987. Iris: An object-oriented database management system. *ACM Trans. on Office Information Systems* 5(1):48–69.

George Gottlob; Michael Schrefl; and Brigitte Röck. 1996. Extending Object-Oriented Systems with Roles. *ACM Transactions on Information Systems* 14(3):268–296.

G.Kappel; W.Retschitzegger; and W.Schwinger. 1998. A Comparison of Role Mechanisms in Object-Oriented Modeling. In *Proceedings of Modellierung'98, K. Pohl, A. Schurr, G. Vossen (eds), Bericht Nr. 6/98-I, Angewandte Mathematik und Informatik, Universitat Munster*, 105–109.

Grady Booch; James Rumbaugh; and Ivar Jacobson. 2000. *The unified modeling language user guide*. Addison Wesley.

Guy Genilloud, and Alain Wegman. 2000a. A foundation for the concept of role in object modelling. In *The 4th International Enterprise Distributed Object Computing Conference*.

Guy Genilloud, and Alain Wegman. 2000b. A new foundation for the concept of roles, and why it makes sense. In *Proceedings of the Ninth International OOPSLA Workshop on Behavioral Semantics*.

Joel Richardson, and Peter Schwarz. 1991. Aspects: Extending Objects to Support Multiple, Independent Roles. In *Proceedings of the ACM SIGMOD Int. Con. on Management of Data*, volume 20, 298–307.

John J. Shilling, and Peter F.Sweeney. 1989. Three Steps to Views: Extending the Object Oriented Paradigm. *Proceedings of OOPSLA* 24(10):352–361.

Li, Q., and R. K. Wong. 1999. Multifaceted object modeling with roles: A comprehensive approach. *Information Sciences* 117:243–266.

Liping Zhao, and Ted Foster. 1999. Modeling Roles with Cascade. *IEEE Software* 86–93.

Martin F. Fowler. 1997. Dealing with Roles. In *Proceedings of the 4th Annual Conference on the Pattern Languages of Programs*, volume 35.

Martin, R.,; D. Riehle; and F. Buschmann. 1998. *Pattern languages of program design 3*. Addison Wesley.

Mehmet Aksit. 1996. Composition and separation of concerns in the object-oriented model. *ACM Computing Surveys* 28A(4).

Peter Coad. 1992. Object-Oriented Patterns. *Communications of the ACM* 35(9):153–159.

P.Papazoglou, and B.J.Kramer. 1997. A Database Model for Object Dynamics. *The VLDB Journal* 6:73–96.

Raymond K. Wong; H.Lewis Chau; and Frederik H.Lochovsky. 1997. A Data Model and Semantics of Objects with Dynamic Roles. In *Proceedings of the 13th International Conference on Data Engineering, IEEE CS Press, University of Bermingham*, 402–411.

Raymond K. Wong. 1999. Heterogeneous and Multifaceted Multimedia Objects in DOOR/MM: A Role-Based Approach with Views. *Journal of Parallel and Distributed Computing* 251–277.

Rushikesh K. Joshi; N.Vivekananda; and D.Janaki Ram. 1997. Message Filters for Object-Oriented Systems. *Software-Practice and Experience* 27(6):677–699.

Steimann, F. 2000. On the representation of roles in object-oriented and conceptual modelling. *Data Knowledge Engineering* 35(1):83–106.

Tamai, T. 1999. Objects and roles: Modeling based on the dualistic view. *Information and Software Technology* 41(14):1005–1010.

Trygve Reenskaug. 1995. *Working with Objects: The OORAM Software Engineering Method*. Manning PublicationsGreenwich, CT.

Upakaram Gopal, and Chandra Sekharaiah K. 2003. Schizophrenia in System Modeling. In *Second International Conference on Document Analysis and Recognition*, 1–5.