Object-oriented Database System Design - Hiroshi Ishikawa - 2012-12-06

Computer Science Workbench is a monograph series which will provide you with an in-depth working knowledge of current developments in computer technology. Every volume in this series will deal with a topic of importance in computer science and elaborate on how you yourself can build systems related to the main theme. You will be able to develop a variety of systems, including computer software tools, computer graphics, database management systems, and computer-aided design and manufacturing systems. Computer Science Workbench represents an important contribution to the field of computer technology. Tunyaj M. Kunju Preface The goal of this book is to give concrete answers to questions such as what object-oriented databases are, why they are needed, how they are designed, and how they are applied. This book focuses on the object-oriented database system called Jasmine. That is, this book aims at creating a concise view of what object-oriented databases are. The contents of this book are directly related to the results of the Jasmine project conducted at Fujitsu Laboratories, Ltd. The book is a polished version of my doctoral dissertation, which includes research papers which I have authored and published.

Object-oriented Database System Design Clearly Explained - Jan L. Harrington - 2001

Object-oriented database systems have been approached with mainly two major intentions in mind, namely to better support new application areas including CAD/CAM, and how to design and implement one. The handbook will benefit database analysts, database administrators, programmers and members of technical staff interested in data models. Andleigh is the author of UNIX SYSTEM ARCHITECTURE. This monograph presents the fundamentals of object databases, with a specific focus on conceptual modeling of object database designs. After an introduction to the fundamental concepts of object-oriented database design, the monograph provides a review of object-oriented conceptual modeling techniques using half-side-by-side Enhanced Entity Relationship diagrams and Unified Modeling Language conceptual class diagrams that feature class hierarchies with specialization constraints and object associations. These object-oriented conceptual models provide the basis for introducing case studies that illustrate the use of object features within the design of object-oriented and object-relational databases. For the object-oriented database perspective, the Object Data Management Group data definition language provides a portable, language-independent specification of an object schema, together with an SQL-like object query language. LinQ (Language Integrated Query) is presented as a case study of an object query language together with its use in the db2 opensource object-oriented database. For the object-relational perspective, the object-relational features of the SQL standard are presented together with an accompanying case study of the object-relational features of Oracle. For completeness of coverage, an appendix provides a mapping of object-oriented conceptual designs to the relational model and its associated constraints. &apos;&apos;[4] of cov.

Object-oriented Database System Design and Implementation For Advanced Applications - Michael Blaha - 1998

This book refines, extends and enhances the general Object Modeling Technique (OMT) methodology for the specific subject matter of database applications. By refining and extending the OMT methodology, the authors provide a set of guidelines and elaborate upon the appropriate methodological steps. The authors present a uniform treatment that addresses files, relational databases, and object-oriented databases.

Object-oriented Modeling and Design for Database Applications - Michael Blaha - 1998

This book refines, extends and enhances the general Object Modeling Technique (OMT) methodology for the specific subject matter of database applications. By restricting the scope of coverage, the authors are able to present more focused examples and elaborate upon the appropriate methodological steps. The authors present a uniform treatment that addresses files, relational databases, and object-oriented databases.

Object-oriented Modeling and Design for Database Applications - Michael Blaha - 1998

This book refines, extends and enhances the general Object Modeling Technique (OMT) methodology for the specific subject matter of database applications. By restricting the scope of coverage, the authors are able to present more focused examples and elaborate upon the appropriate methodological steps. The authors present a uniform treatment that addresses files, relational databases, and object-oriented databases.

Object-Oriented Database Systems. Directions for Future Research and Development. - Michael Blaha - 1998

This book presents the fundamentals of object databases, with a specific focus on conceptual modeling of object database designs. After an introduction to the fundamental concepts of object-oriented database design, the monograph provides a review of object-oriented conceptual modeling techniques using half-side-by-side Enhanced Entity Relationship diagrams and Unified Modeling Language conceptual class diagrams that feature class hierarchies with specialization constraints and object associations. These object-oriented conceptual models provide the basis for introducing case studies that illustrate the use of object features within the design of object-oriented and object-relational databases. For the object-oriented database perspective, the Object Data Management Group data definition language provides a portable, language-independent specification of an object schema, together with an SQL-like object query language. LinQ (Language Integrated Query) is presented as a case study of an object query language together with its use in the db2 opensource object-oriented database. For the object-relational perspective, the object-relational features of the SQL standard are presented together with an accompanying case study of the object-relational features of Oracle. For completeness of coverage, an appendix provides a mapping of object-oriented conceptual designs to the relational model and its associated constraints. &apos;&apos;[4] of cov.

Object-Oriented Database Systems. Directions for Future Research and Development. - Michael Blaha - 1998

This book presents the fundamentals of object databases, with a specific focus on conceptual modeling of object database designs. After an introduction to the fundamental concepts of object-oriented database design, the monograph provides a review of object-oriented conceptual modeling techniques using half-side-by-side Enhanced Entity Relationship diagrams and Unified Modeling Language conceptual class diagrams that feature class hierarchies with specialization constraints and object associations. These object-oriented conceptual models provide the basis for introducing case studies that illustrate the use of object features within the design of object-oriented and object-relational databases. For the object-oriented database perspective, the Object Data Management Group data definition language provides a portable, language-independent specification of an object schema, together with an SQL-like object query language. LinQ (Language Integrated Query) is presented as a case study of an object query language together with its use in the db2 opensource object-oriented database. For the object-relational perspective, the object-relational features of the SQL standard are presented together with an accompanying case study of the object-relational features of Oracle. For completeness of coverage, an appendix provides a mapping of object-oriented conceptual designs to the relational model and its associated constraints. &apos;&apos;[4] of cov.

Distributed Object-oriented Data-systems Design - Prabhat K. Andleigh - 1992

This guide deals with an introduction to the design and implementation of advanced information systems. It covers object-oriented database management systems, distributed environments, and advanced user interfaces i.e. those integrating text, pictures, video and sound. This book also focuses on migration issues involved in going from relational database management systems to object-oriented database management systems, and discusses the advantages/disadvantages of both types of systems. The authors have developed a unique Frame-Object Analysis Methodology for advanced modeling. It also shows that the Frame-Object concept is key to providing a consistent and comprehensive understanding of developments in this flourishing area of current database research. Data model and language aspects, interface and database design issues, architectural and implementation questions are covered. Although based on a series of workshops, the contents of this book has been carefully edited to reflect the current state of international research in object oriented database design and implementation.

Distributed Object-oriented Data-systems Design - Prabhat K. Andleigh - 1992

This guide deals with an introduction to the design and implementation of advanced information systems. It covers object-oriented database management systems, distributed environments, and advanced user interfaces i.e. those integrating text, pictures, video and sound. This book also focuses on migration issues involved in going from relational database management systems to object-oriented database management systems, and discusses the advantages/disadvantages of both types of systems. The authors have developed a unique Frame-Object Analysis Methodology for advanced modeling. It also shows that the Frame-Object concept is key to providing a consistent and comprehensive understanding of developments in this flourishing area of current database research. Data model and language aspects, interface and database design issues, architectural and implementation questions are covered. Although based on a series of workshops, the contents of this book has been carefully edited to reflect the current state of international research in object oriented database design and implementation.
Object-oriented databases are increasingly relying on an object-oriented approach as the best way to meet user needs and performance criteria. This book teaches you how to use the Unified Modeling Language (UML) to model object-oriented design models, and how to translate UML models into database components. It explains how to use database design concepts to transform a design into a relational, object-oriented, and object-relational databases for all major database management systems. Oracle Database 11g gives you the tools you need to design, build, and test databases using an OO model. The book shows how to use UML, the accepted standard for database design according to OGC principles. Explains how to transform your designs into a conceptual schema for relational, object-relational, and object-oriented systems. Offers practical examples of design for Oracle, SQL Server, Sybase, Informix, Object Design, POET, and other database management systems. Focuses heavily on reusing design patterns for maximum productivity and teaches you how to certify completed designs for reuse.

Object-Oriented Database Systems - Robert J. Muller - 1999-03-08


Object-oriented database management systems (ODBSMS) have generated significant excitement in the database community in the last decade. This interest stems from two main contributions of object-oriented technology: the effective management of complex objects, and the improvements over relational systems in meeting the requirements of these complex objects. This book examines both these contributions of object-oriented technology, and illustrates them through the design process, from requirements analysis to schema generation. You’ll learn to express stakeholder needs in UML use case and actor diagrams, to translate UML entities into database components, and to transform the resulting design into relational, object-relational, and object-schemas for all major database management systems. Oracle Database 11g gives you the tools you need to design, build, and test databases using an OO model. The book shows how to use UML, the accepted standard for database design according to OGC principles. Explains how to transform your designs into a conceptual schema for relational, object-relational, and object-oriented systems. Offers practical examples of design for Oracle, SQL Server, Sybase, Informix, Object Design, POET, and other database management systems. Focuses heavily on reusing design patterns for maximum productivity and teaches you how to certify completed designs for reuse.

Object-Oriented Database Systems - Robert J. Muller - 1999-03-08


Object-oriented database management systems (ODBSMS) have generated significant excitement in the database community in the last decade. This interest stems from two main contributions of object-oriented technology: the effective management of complex objects, and the improvements over relational systems in meeting the requirements of these complex objects. This book examines both these contributions of object-oriented technology, and illustrates them through the design process, from requirements analysis to schema generation. You’ll learn to express stakeholder needs in UML use case and actor diagrams, to translate UML entities into database components, and to transform the resulting design into relational, object-relational, and object-schemas for all major database management systems. Oracle Database 11g gives you the tools you need to design, build, and test databases using an OO model. The book shows how to use UML, the accepted standard for database design according to OGC principles. Explains how to transform your designs into a conceptual schema for relational, object-relational, and object-oriented systems. Offers practical examples of design for Oracle, SQL Server, Sybase, Informix, Object Design, POET, and other database management systems. Focuses heavily on reusing design patterns for maximum productivity and teaches you how to certify completed designs for reuse.

Object-Oriented Database Systems - Robert J. Muller - 1999-03-08


Object-oriented database management systems (ODBSMS) have generated significant excitement in the database community in the last decade. This interest stems from two main contributions of object-oriented technology: the effective management of complex objects, and the improvements over relational systems in meeting the requirements of these complex objects. This book examines both these contributions of object-oriented technology, and illustrates them through the design process, from requirements analysis to schema generation. You’ll learn to express stakeholder needs in UML use case and actor diagrams, to translate UML entities into database components, and to transform the resulting design into relational, object-relational, and object-schemas for all major database management systems. Oracle Database 11g gives you the tools you need to design, build, and test databases using an OO model. The book shows how to use UML, the accepted standard for database design according to OGC principles. Explains how to transform your designs into a conceptual schema for relational, object-relational, and object-oriented systems. Offers practical examples of design for Oracle, SQL Server, Sybase, Informix, Object Design, POET, and other database management systems. Focuses heavily on reusing design patterns for maximum productivity and teaches you how to certify completed designs for reuse.

Object-Oriented Database Systems - Robert J. Muller - 1999-03-08


Object-oriented database management systems (ODBSMS) have generated significant excitement in the database community in the last decade. This interest stems from two main contributions of object-oriented technology: the effective management of complex objects, and the improvements over relational systems in meeting the requirements of these complex objects. This book examines both these contributions of object-oriented technology, and illustrates them through the design process, from requirements analysis to schema generation. You’ll learn to express stakeholder needs in UML use case and actor diagrams, to translate UML entities into database components, and to transform the resulting design into relational, object-relational, and object-schemas for all major database management systems. Oracle Database 11g gives you the tools you need to design, build, and test databases using an OO model. The book shows how to use UML, the accepted standard for database design according to OGC principles. Explains how to transform your designs into a conceptual schema for relational, object-relational, and object-oriented systems. Offers practical examples of design for Oracle, SQL Server, Sybase, Informix, Object Design, POET, and other database management systems. Focuses heavily on reusing design patterns for maximum productivity and teaches you how to certify completed designs for reuse.

Object-Oriented Database Systems - Robert J. Muller - 1999-03-08


Object-oriented database management systems (ODBSMS) have generated significant excitement in the database community in the last decade. This interest stems from two main contributions of object-oriented technology: the effective management of complex objects, and the improvements over relational systems in meeting the requirements of these complex objects. This book examines both these contributions of object-oriented technology, and illustrates them through the design process, from requirements analysis to schema generation. You’ll learn to express stakeholder needs in UML use case and actor diagrams, to translate UML entities into database components, and to transform the resulting design into relational, object-relational, and object-schemas for all major database management systems. Oracle Database 11g gives you the tools you need to design, build, and test databases using an OO model. The book shows how to use UML, the accepted standard for database design according to OGC principles. Explains how to transform your designs into a conceptual schema for relational, object-relational, and object-oriented systems. Offers practical examples of design for Oracle, SQL Server, Sybase, Informix, Object Design, POET, and other database management systems. Focuses heavily on reusing design patterns for maximum productivity and teaches you how to certify completed designs for reuse.
Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management system design - Yuni Lio - 1994

Object-oriented database management system design - Yuni Lio - 1994

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)

Object-oriented database management systems (OODBMSs) have generated significant excitement in the database community in the last decade. This interest stems with nested objects (e.g., a vehicle object containing an engine object). Different real-world objects, but the semantics of this association is not part of the database. Furthermore, the attributes of a relation may come only from simple and fixed data types (numeric, character, and, sometimes, data types). Advanced approaches require explicit storage and manipulation of more abstract types (e.g., images, design documents) and the ability for the users to define their own application-specific types. Therefore, a rich type system supporting user-defined abstract types is required. The ODBMS must be relatively simple and flat manner. Non-traditional approaches require more complex object structures with nested objects (e.g., a vehicle object containing an engine object)
"Development and Implementation of an Object-Oriented Database Management System," which spans the programming and database worlds by integrating procedural and representational capability and the requirements of multi-user persistent storage and the two chapters on object-oriented environments provide a representative sample of good research in these two important areas. Bruce Silver is a researcher at IBM's Thomas J. Watson Research Center. Peter Wegner is a professor in the Department of Computer Science at Brown University. Research Directions in Object-Oriented Programming is included in the Computer Systems series, edited by Herb Schorrman.

Advances in Object-oriented Data Modeling - M. Papazoglou - 2000
This book focuses on recent developments in representational and processing aspects of complex data-intensive applications. Until recently, information systems have been designed around different business functions, such as accounts payable and inventory control. Object-oriented modeling, in contrast, structures systems around the data—the objects—that make up the various business functions. Because information about a particular function is limited to one place—to the object—the system is shielded from the effects of change. Object-oriented modeling also promotes better understanding of requirements, clear designs, and more easily maintainable systems. This book focuses on recent developments in representational and processing aspects of complex data-intensive applications. The chapters cover "hot" topics such as application behavior and consistency, reverse engineering, interoperability and collaboration between objects, and work-flow modeling. Each chapter contains a review of its subject, followed by object-oriented modeling techniques and methodologies that can be applied to real-life applications. Contributors: F. Casati, S. Ceri, R. Cicchetti, L. M. Delcambre, E. F. Eckland, D. W. Embley, G. Engels, J. M. Gagnon, S. R. M. Gogolla, L. Groenewegen, C. S. Jensen, G. Kappel, B. J. Krämer, S. W. Liddell, R. Missaoui, M. Norrie, M. P. Papazoglou, C. Parent, B. Pernici, P. Poncelet, G. Pozzi, M. Schmidt, R. T. Snoeijers, S. Spaccapietra, M. Stumptner, M. Tenenbaumer, W. J. van den Heuvel, S. N. Woodfield.

Advances in Object-oriented Data Modeling - M. Papazoglou - 2000
This book focuses on recent developments in representational and processing aspects of complex data-intensive applications. Until recently, information systems have been designed around different business functions, such as accounts payable and inventory control. Object-oriented modeling, in contrast, structures systems around the data—the objects—that make up the various business functions. Because information about a particular function is limited to one place—to the object—the system is shielded from the effects of change. Object-oriented modeling also promotes better understanding of requirements, clear designs, and more easily maintainable systems. This book focuses on recent developments in representational and processing aspects of complex data-intensive applications. The chapters cover "hot" topics such as application behavior and consistency, reverse engineering, interoperability and collaboration between objects, and work-flow modeling. Each chapter contains a review of its subject, followed by object-oriented modeling techniques and methodologies that can be applied to real-life applications. Contributors: F. Casati, S. Ceri, R. Cicchetti, L. M. Delcambre, E. F. Eckland, D. W. Embley, G. Engels, J. M. Gagnon, S. R. M. Gogolla, L. Groenewegen, C. S. Jensen, G. Kappel, B. J. Krämer, S. W. Liddell, R. Missaoui, M. Norrie, M. P. Papazoglou, C. Parent, B. Pernici, P. Poncelet, G. Pozzi, M. Schmidt, R. T. Snoeijers, S. Spaccapietra, M. Stumptner, M. Tenenbaumer, W. J. van den Heuvel, S. N. Woodfield.

Practical Application of Object-Oriented Techniques to Relational Databases - Donald K. Burleson - 1994-03-16
This highly practical book shows systems professionals how to apply object-oriented techniques to relational databases immediately. Burleson demonstrates approaches that systems engineers use to put object methods into design and development process immediately. Moreover, it is not clear whether object-oriented database systems would be superior than relational database systems for supporting real-time applications. In this paper, we address issues that must be investigated in order to design and develop an object-oriented database system for real-time applications. Also, we present a model that integrates features for scheduling real-time transactions with the traditional object-oriented database model.

Issues in Developing Object-Oriented Database Systems for Real-Time Applications - 1994
Database systems for real-time applications must satisfy timing constraints associated with transactions, in addition to maintaining data consistency. Recently, interest in object-oriented databases has been growing for nontraditional applications of database systems, and several real-time applications are being developed using an object-oriented database system. Object-oriented database systems would be superior than relational database systems for supporting real-time applications. In this paper, we address issues that must be investigated in order to design and develop an object-oriented database system for real-time applications. Also, we present a model that integrates features for scheduling real-time transactions with the traditional object-oriented database model.

Issues in Developing Object-Oriented Database Systems for Real-Time Applications - 1994
...