Journal of Software Engineering and Applications, **2011, 4, 666-671**
doi:10.4236/jsea.2011.412078 Published Online December 2011 (http://www.SciRP.org/journal/jsea)

Developing Systems by Using Object Oriented Database Practical Study on ISO 9001:2000 System

Kamel Khoualdi[1], Thoria Alghamdi[2]

[1]Management Information Systems Department, King Abdulaziz University, Jeddah, Saudi Arabia; [2]Information Systems Department, King Abdulaziz University, Jeddah, Saudi Arabia.
Email: {kamel_khoualdi, thalghamdi}@yahoo.com

Received October 15th, 2011; revised November 20th, 2011; accepted November 30th, 2011.

ABSTRACT

Object oriented database OODB is the third generation of databases. It was a natural result of limitations on relational database and increasing requirements of software, and business needs. This paper presents the method to apply OODB on ISO 9001:2000 *System. It focuses in analysis and design phase by using Unified Modeling Language UML, and how can represent this type of database. This paper provides comprehensive information about applying OODB on software.*

Keywords: *ISO* 9001:2000, *Object Oriented Database, Relational Database, UML*

1. Introduction

In today's world, applications become more complex and require high performance. These requirements are difficult to manage in traditional Relational Database Management System RDBMS so organizations of all sizes are attempting to employ database technology to meet these challenges [1]. Object Oriented Database OODB is new model of database; it is created to avoid the limitations encountered in the past models. The mechanism of OODB depends on object concept, in this concept data is stored in the database as object, each object has its state and behavior, the state is defined by the value of its properties, and behavior is defined by the method that operates on the state of the object [2]. Object oriented technique supports three concepts which represent the source of OODB power, these concepts are: Inheritance Classes can inherit the attributes and behaviors of other classes. They represent a hierarchy structure. Inheritance develops complex software by creating new objects and allowing it to hold new features in addition to all old features [3]. Polymorphism it allows sharing specification of the operation with other objects, these objects can further extend this operation to provide behaviors that are unique to those objects [4]. Encapsulation the manipulation of an object is possible only through its defined external interface; the instance variables and methods are hidden.

2. ODBMS Advantages

OODBMS uses Object to represent data. It can represent real world and complex relationships, and it can represent a hierarchical structure. It is also able to develop systems faster than RDBMS by using inheritance. Object structure supports Encapsulation, concurrency, and ad hoc query [1]. Also, it can store more data like images, video, audio, animations and mixed media, and accessing data can be faster because objects can be retrieved directly by following pointers. It use one UML diagram and support inverse relationships [5]. It supports an Object Identifier OID that is automatically generated by the system. This mechanism guarantees uniqueness to each object, an OID cannot be modified by the application, and this way eliminates the need for user defined keys in the OODB model. In OODB model, developer can add any rules by writing codes in one or more functions [6]. OODB decreases the gap between programming in Object Oriented Language and accessing data from database.

3. Class Diagram

ISO 9001:2000 System database was designed by using object data type to represent each component in the manual system such as employee, unit, document, TQM manual, meeting, training file, auditing and corrective action.

The employee class inherits all the primary data from the person class and it contains a data of account type. The employee class and unit class have an association relationship, where one or more employees belong to one unit, and one unit is managed by one employee. The employee class and the committee class have an association relationship, where one employee has zero or more committees. **Figure 1** shows relationship between the employee class and other classes. In Code.1 the first SQL sentence shows the implementation of employee Class and the second sentence shows the implementation of employee table where each row in the table is object of an employee type.

```
SQL > CREATE OR REPLACE TYPE employee_type
UNDER person_typ
(id              INTEGER,
j_name           VARCHAR2(30),
j_degree         VARCHARE2(30),
salary           INTEGER,
education_level  VARCHAR2(30),
h_date           DATE,
l_data           DATE,
unit        REF  unit_typ,
account          account_type,
committee   REF  committee_typ);
SQL > CREATE TABLE employee_tab of employee typ
(PRIMARY KEY (id));
```

Also, the database contains document as a parent class; the document can be internal document if it was issued inside organization or external document if not, internal documents include decisions, forms, check lists, reports, and TQM manual documents. The document class and the unit class have an association relationship, where each document belongs to a specific unit. The document-class and the employee class have an association relationship, where each document was written or approved by one employee. **Figure 2** shows the document class and its relationship.

```
SQL > CREATE OR REPLACE TYPE document_typ
AS OBJECT
```

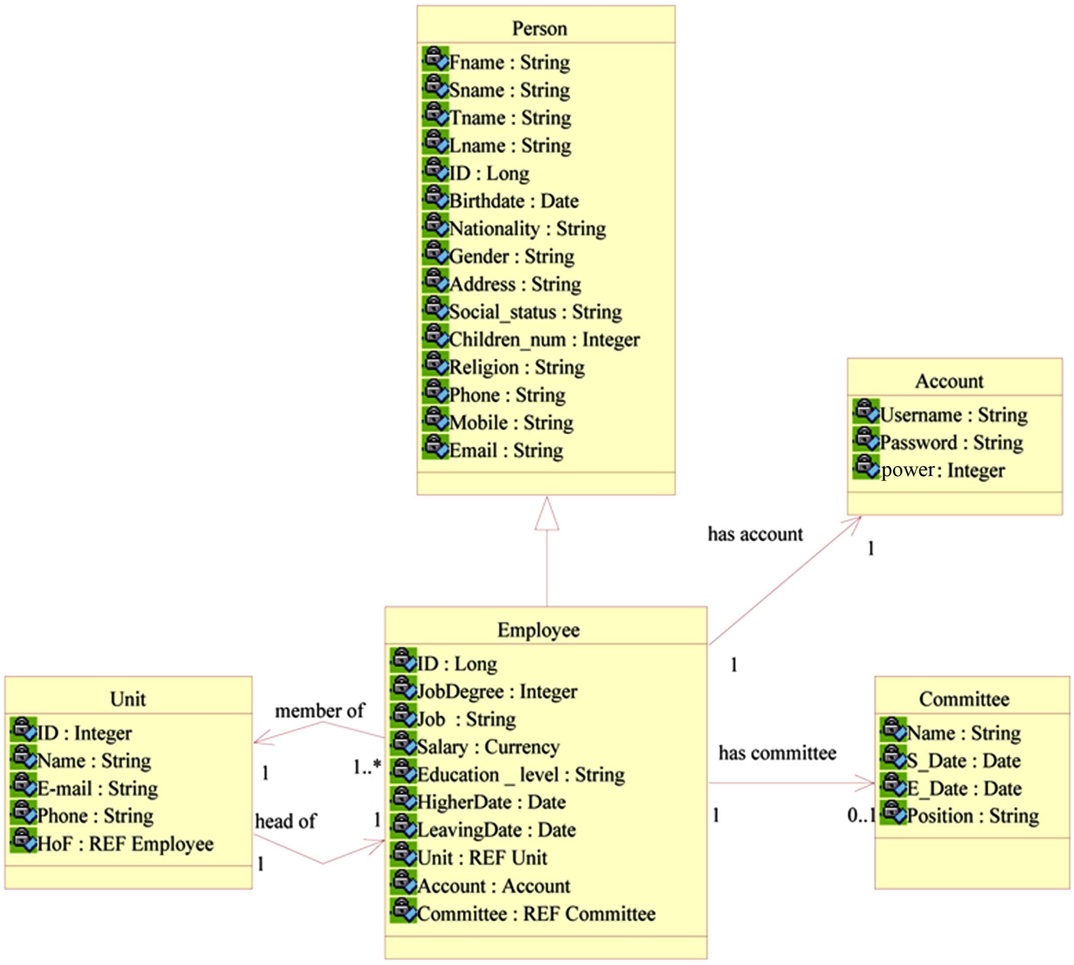

Figure 1. Relationships between employee class and other classes.

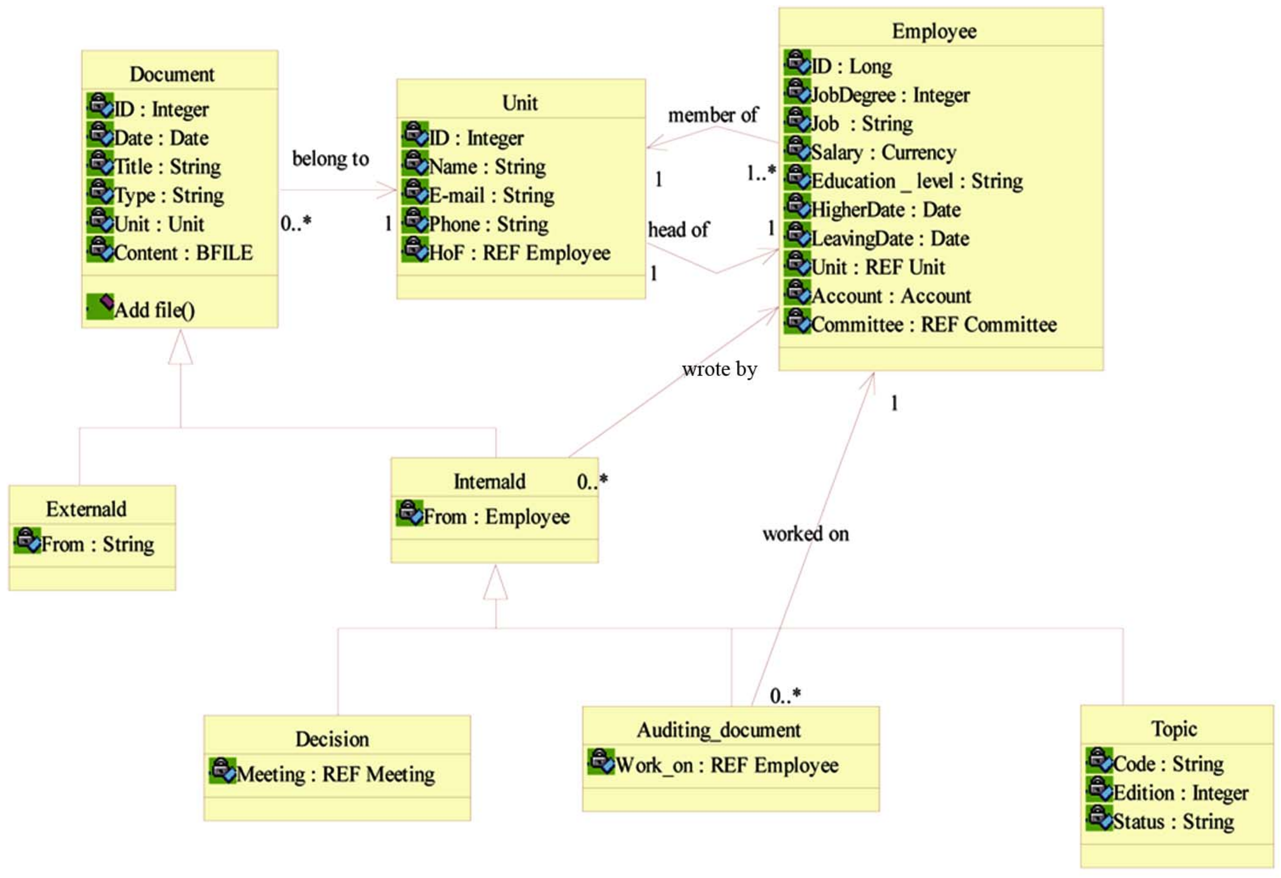

Figure 2. Document class and its relationship.

```
(id        INTEGER,
Ddate      DATE,
title      VARCHAR2(50),
type       VARCHAR2(15),
unit       REF     unit_typ,
content    BFILE)  NOT FINAL;
```

SQL > CREAT OR REPLACE TYPE internaldoc_typ UNDER document_typ

(*Employ_name REF employee_typ*) *NOT FINAL*;

SQL > CREATE TABLE internaldoc_tab of internal typ (*PRIMARY KEY* (*id*));

In Code.2 the first SQL sentence shows the implementation of document class. We set the type of content as BFILE. By using this type we can store the extension of the file in the database and store the file (DOC, PTT, PDF) in separated folder. The NOT FINAL sentence refers to the ability to inherit all data and function from this class. The second SQL sentence shows the implementation of internal document as a child class. It has only one data it pointes to Employee object. We use this because the internal document should be written by an employee.

The TQM manual was represented as a class contains unlimited number of parts, and each part contains unlimited number of chapters, and each chapter contains unlimited number of subjects, and each subject refers to special topic, so we use the aggregation relationship between the TQM manual and the part class, and so on. We use the association relationship between the chapter class and the topic class instead of aggregation because the topic class should be associate to other classes such as corrective and request classes, and by using association relationship it will be easy to make any type of corrective action in TQM manual, and general manager can reject any TQM manual document without lose it. **Figure 3** shows the TQM manual class.

The Database of this system was designed depends on view which is looking to ISO system as a set of actions related with special documents. Each action needs specific information like the date, the responsible employee, the title and the status. For example: the meeting is an action generates documents of decision type, the auditing is an action generates chick list and report, documentation request is an action need special form and generates TQM manual document, corrective action is an action need special form and in some case it generates a new

TQM manual document, and so on. Depends on this view the database was designed to be contain action class as parent class and the meeting class, the request class, and the auditing class as a children classes. The corrective action class and the documentation request class are children of the request class, because they need an application. **Figure 4** shows the class diagram for ISO 9004: 2000 System.

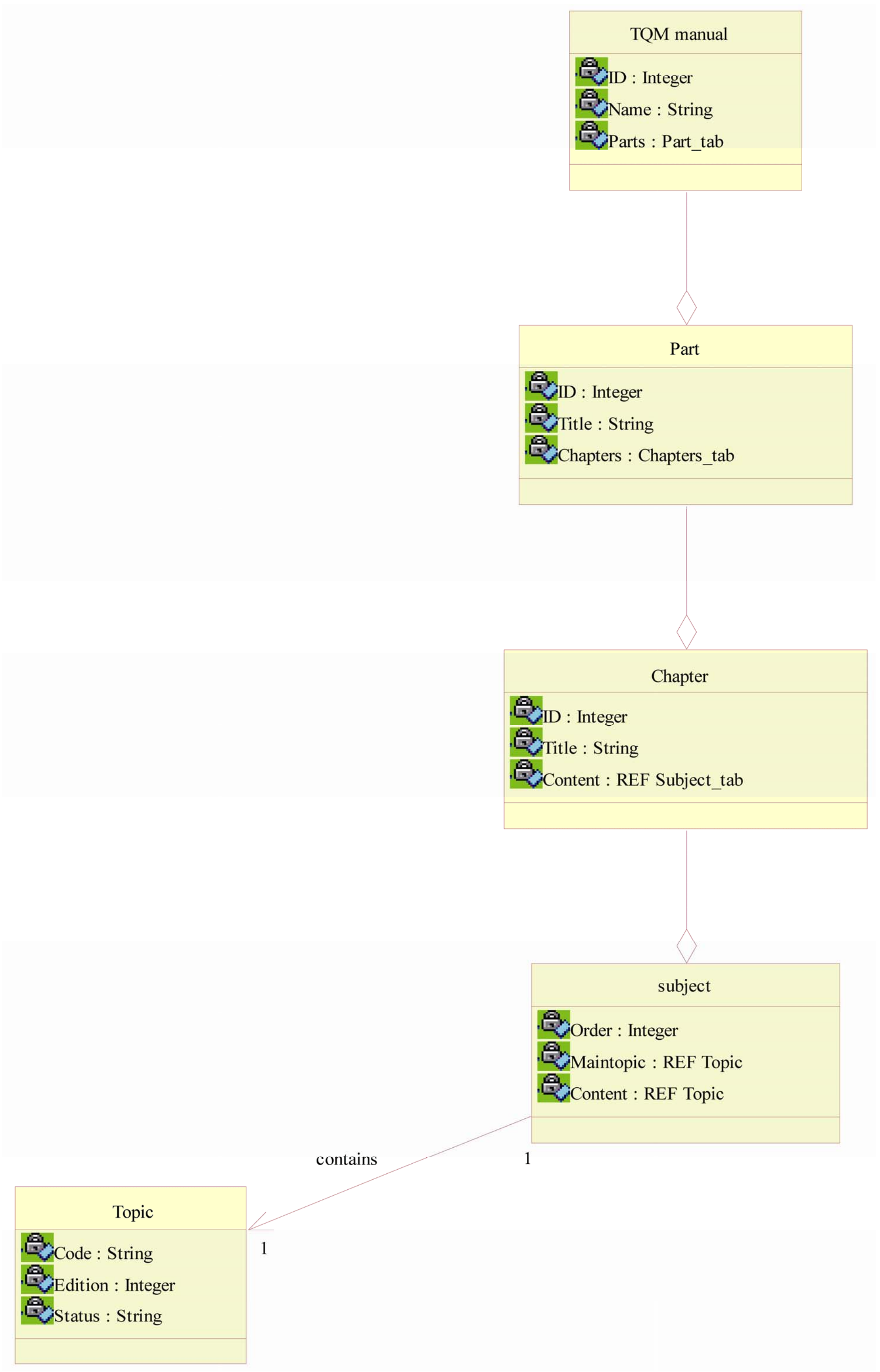

Figure 3. The TQM manual class.

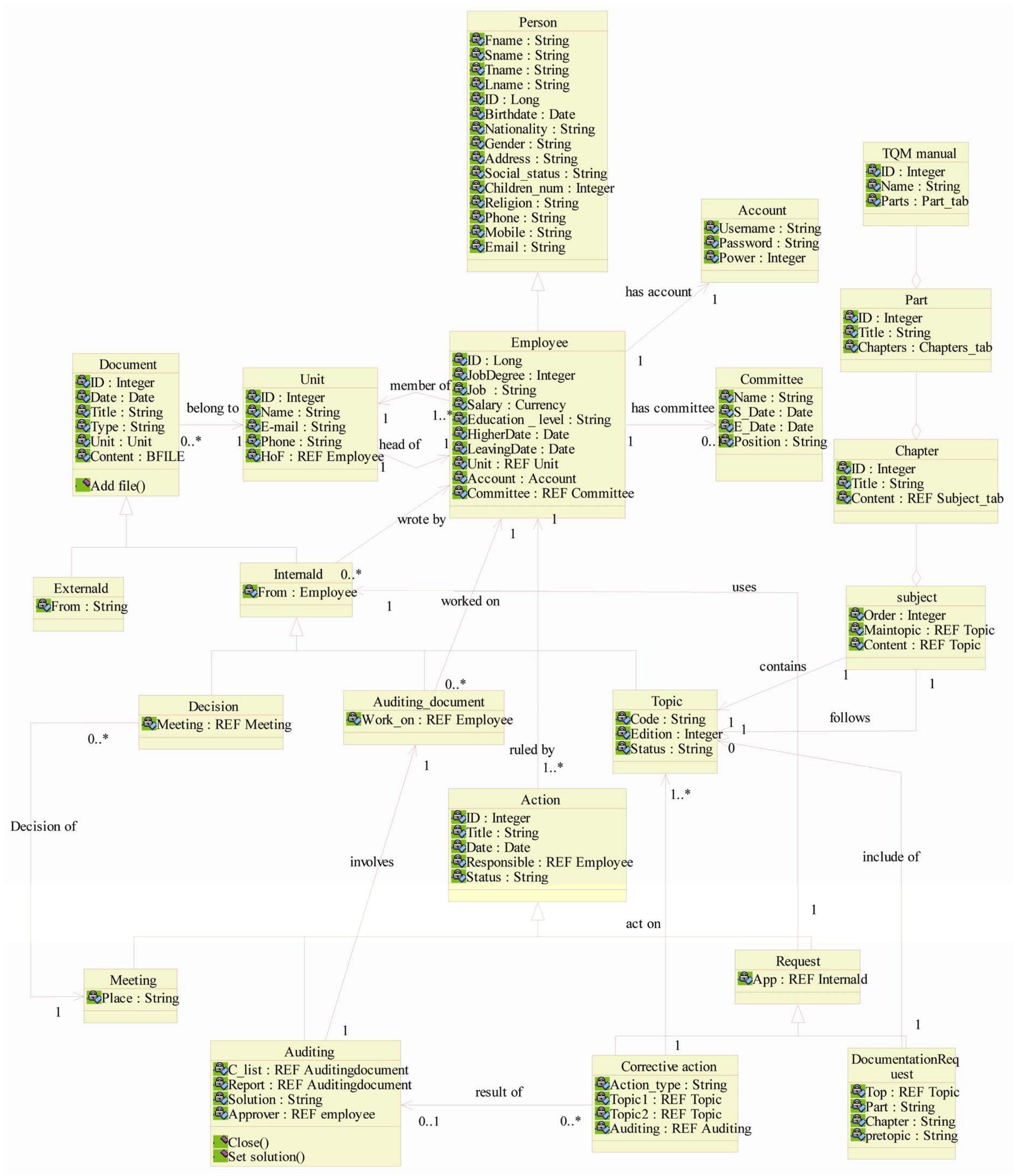

Figure 4. IS0 9001:2000 System marked as class diagram.

4. Conclusions

Our objective in this thesis is to develop a database for ISO 9001:2000 System by using Object Oriented Approach. We studied the actual manual ISO 9001:2000 System, and depend on the system's requirements and components, we generated class diagrams by using UML. In particular, the Object Oriented Database can represent and handle the complex data in a high performance and can develop the systems faster with little code. We can also note that modeling database of ISO system in OODB looks simple and easy to develop and maintenance.

REFERENCES

[1] S. Luthra, "Architecture in Object Oriented Databases,"

2008. http://www.rimtengg.com/coit2007/proceedings/pdfs/63.pdf

[2] S. Bagui, "Achievements and Weaknesses of Object-Oriented Databases," *Journal of Object Technology*, Vol. 2, No. 4, 2003, pp. 29-41. doi:10.5381/jot.2003.2.4.c2

[3] C. Chambers, D. Ungar, B. Chang and U. Hölzle, "Parents Are Shared Parts of Objects: Inheritance and Encapsulation in Self," *Lisp and Symbolic Computation*: *An International Journal*, Vol. 4, No. 3, 1991, pp. 207-222.

[4] S. Neelam and T. Benjamin, "Testing Polymorphic Behavior," *Journal of Object Technology*, Vol. 1, No. 3, 2002, pp. 173-188. doi:10.5381/jot.2002.1.3.a10

[5] D. Obasanjo, "An Exploration of Object Oriented Database Management Systems," 2001. www.25hoursaday.com/WhyArentYouUsingAnOODBMS.html

[6] M. Stonebraker, L. A. Rowe, B. G. Lindsay, J. Gray, M. J. Carey and D. Beech, "The Committee for Advanced DBMS Function: Third Generation Data Base System Manifesto," *SIGMOD Conference*, Vol. 19, No. 3, 1990, pp. 495-511.