An Open Architecture for Medical Imaging Systems

William Chimiak and Daniel Lopez

Integration of medical imaging is the goal of many standards. Interfile and Digital Imaging and Communications in Medicine (DICOM) standards have made great strides in this area. The impact of DICOM is greater owing to its wider acceptance. However, the lack of conformance testing makes interoperability difficult, often requiring expensive consulting teams in a mixed-vendor environment. DICOM is difficult to learn and deploy because it does not use modern software paradigms. Many hardware and software advances in computer architectures and communications could greatly enhance diagnostic imaging if they could be quickly incorporated into medical imaging standards. These problems led to consideration of a solution able to take advantage of state-of-the-art technology and to allow diverse implementations while ensuring the accurate interchange of information between health care providers. It is an object-oriented system information model, with the common object request broker architecture (CORBA) as the vehicle to ensure interoperability between different implementations of the model. This report presents the model.

Copyright © 2000 by W.B. Saunders Company

KEY WORDS: medical imaging, information model, CORBA, DICOM, interfile.

THIS ARTICLE proposes a software component system based on the common object request broker architecture (CORBA) to handle images. The Digital Imaging and Communications in Medicine (DICOM)¹ would operate under it. The proposed system is called the Medical Imaging Information Model or MIIM. The MIIM uses CORBA,²⁻⁴ which allows rapid deployment of new technologies to the medical imaging information model easily, unlike DICOM. MIIM also contains the framework to incorporate legacy systems in a standard manner, and CORBA uses software paradigms found in the many horizontal markets.

From Wake Forest University School of Medicine, Winston-Salem, NC.

The work on this project by Daniel Lopez was funded by the postgraduate program for extended studies of "Sa Nostra" Caixa de Balears, Balears, Spain.

Address reprint requests to William Chimiak, PhD, NGI Telemedicine Research Engineer, Center for Health Sciences Communications, East Carolina University, 1S-10 Brody Medical Science Bldg, Greenville, NC 27858-4354.

Copyright © 2000 by W.B. Saunders Company

0897-1889/00/1302-0004\$10.00/0

doi:10.1053/di.2000.5422

The diversity of needs commonly found in health care institutions adds to the complexity of interfacing a PACS to a radiology information system (RIS) and/or a hospital information system (HIS). This diversity also results in different implementations of the same information system at various sites, thereby increasing the difficulty of programming for vendors. Even within the DICOM standard, the lack of conformance testing makes multivendor picture archiving and communication systems (PACS) integration a major effort. This was experienced at Wake Forest University School of Medicine in several such DICOM PACS integration efforts with secondary capture, CT imaging modalities, and service-class user/service-class provider harmonization efforts. This is partly related to the rapid development of new technologies coupled with a broad range of possible individual answers to the same problem. This, in turn, creates obstacles to the interchange of information between different implementations. These problems led us to consider a solution able to take advantage of state-of-the-art technology and allow diverse implementations while ensuring the accurate interchange of information among health care providers.

MAIN GOALS

Although this type of intercommunications infrastructure could be used to solve many different problems, the goals of highest interest in the medical imaging information model (MIIM) are:

1. To ensure interoperability between health care providers so they can share medical images and related information transparently and effortlessly.
2. To maintain interoperability regardless of how each provider stores information.
3. To allow changes in the way this information is stored, without major software changes.
4. To facilitate the creation of applications that access and use this information without forcing the programmers to learn how each provider stores information or to learn protocols and languages different from the ones to which they are accustomed. In this way, the quality and productivity of medical imaging applications would increase.

These goals are an important step in pursuing the

main objective of medical informatics, namely improving the quality of the health care provided while decreasing the costs and facilitating the practice of medicine.

MEANS

To accomplish these goals, a clear and unique model of the medical imaging information to be accessed and interchanged must be defined first. Then, an interface to the imaging information model must be defined to allow access to the information, independent of how the model is implemented and independent of the language from which the information is retrieved. A convenient way to implement the model must be provided for the medical information system developers that maintains the independence of access to the information from a programming point of view. Access restrictions and security issues also must be addressed.

The interface must be written so that it can be easily extended and able to evolve as the medical imaging information model does. By writing the interface this way, the sources of medical information are detached from the programs that make use of the information. This will accomplish the following:

1. Ease the development of medical imaging applications, independent of the implementation programming language. C++, popular for speed and modularity, and the ubiquitous Java language are at least two programming languages to be considered.
2. Allow the implementation of the information systems following different internal models without losing interoperability between different systems and applications. For example, one system may be based on an object-oriented database, another on an SQL database, and a third on a JAVA database control (JDBC) activated database. These three should easily intercommunicate within this interface.
3. Create the framework for a high degree of interoperability between sources of medical imaging information and applications. This allows any application to operate with any source and to access any source. Issues such as security are handled elsewhere, reducing software code complications and system complexity with appropriate functional decomposition. For example, the database will have its

access restrictions, and possibly even cryptography. The data transport may be carried on an internet protocol (IP) or virtual private network (VPN) or may even use an Ipsec tunnel, which is encrypted further. It may not have any security in upper layers but a fully encrypted data link layer (such as fully encrypted Ethernet and/or asynchronous transfer mode [ATM] with single or public key encryption with secure periodic or aperiodic key exchanges). The point is, by using MIIM, we will not preclude a system integrator from delivering an interoperable solution with the best available subsystems to solve the system requirements. Security is not part of this component system; it will use security components of the overall system. However, keep in mind, if one employs an MIIM system, it will not operate with another secure MIIM system that does not have the necessary secret keys; compromising the system can be as difficult as the hospital desires, within budget constraints.

The methodology must provide for the interfacing of legacy systems to avoid inordinate costs or forced acceleration of implementation plans.

In summary, the objective of the proposed methodology is not to tell everybody *how* to play, but to specify clearly *the rules* of the game so everybody can play together.

DIGITAL IMAGING AND COMMUNICATIONS IN MEDICINE (DICOM)

The proposed MIIM presented herein is based on and operates under DICOM, version 3, the standard for exchange of digital information between medical imaging equipment.

In 1980, the American College of Radiology (ACR) addressed the problem of transferring images in a hospital by means of a new standard. Early in 1983, the ACR turned to the National Electronics Manufacturing Association (NEMA) as a partner and forum for the development of that standard.

The initial 60-page report recommending slight additions to an existing format, Interfile, which had been successfully used for nuclear medicine images, was dropped. Interfile featured ease of transport as a result of independence of the underlying communications. The ACR-NEMA version 1.0 that emerged from the early standards effort was a

7-layer protocol supporting a point-to-point connection.

A great contribution of the ACR-NEMA committee was the data dictionary provided by radiologists and vendors. This dictionary laid the foundations for the radiology information model and is still a very valuable source of information about the elements involved when dealing with radiological images. Once the data dictionary was completed, the University of Geneva developed a protocol using only the ACR-NEMA header, which includes the image, and allowed the transmission of data to be done by any valid data communications industry standard to allow inexpensive interconnection. The protocol is PAPHYRUS.⁵ Meanwhile, ACR updated the protocol to version 2.0,⁶ a readable 120-page specification with some ambiguities.

In 1990, owing to the dramatic decrease in cost of the networking hardware, the ACR-NEMA committee decided to add support for the OSI protocol while continuing compatibility with the 50-pin connector. Originally, the committee decided not to use TCP/IP because of the feeling that few computers would be running TCP/IP by 1991 and that a majority of computer communications would use the OSI protocol. This opinion was reasonable at the time.⁷ Given that assumption, the ACR-NEMA committee opted to use a solution based on an OSI network control protocol, CMIS/CMIP. This made sense assuming CMIS/CMIP would replace the one used on the Internet, the Simple Network Management Protocol (SNMP).

With version 3.0, ACR-NEMA renamed the protocol as the DICOM protocol. The intention was to move the technology forward, introducing object-oriented concepts. The ACR-NEMA committee was concerned with similarity to CMIS/CMIP creating conflicts with the object-oriented goal. The resulting standard often was confusing and resulted in the unusual *object-based* design instead of a proper *object-oriented* design. One of the major contributions of this version was the clear statement of the basic information model in the DICOM specification, Part 3. This DICOM composite image information object description model is the basis of the information model MIIM uses.

Advantages of DICOM

1. The main advantage of DICOM is that it is used successfully by radiology systems. Before DICOM, only Interfile was capable of

integrating medical images, and it was limited to nuclear medicine. DICOM makes multivendor system intercommunication possible. Modalities, soft copy, and hard copy systems can be made to work together.

2. DICOM now supports TCP/IP, thereby allowing the use of mainstream data communications equipment.
3. The DICOM standard provides an invaluable data dictionary and basic information model that reflect the elements and information involved in medical imaging. These can be used as a starting point for developing the architecture of the information systems designed to manage this kind of data.

Limitations of DICOM

The main limitations of DICOM are as follows:

1. The object-based design of DICOM makes it more difficult to understand. The learning curve that developers must follow before creating DICOM products is not as fast as desired. The necessity of special training for developers working in those systems increases the cost and slows the development cycle. The intent to use object-orientation and the definition of the basic information model are steps in the right direction, although they should also be reflected in the design of the standard.
2. The use of the object-based terminology in the standard makes it confusing to read. Although it appears to deal with objects, it deals with something quite different.
3. The DICOM specification originally was a 120-page document. Systems based on it required additional system integration by consultants in order to approach some level of interoperability. With DICOM, there is now a 22-volume document, and implementations must still have sizable integration efforts to produce working systems. Implementing systems requires that at least two, and most of the time three, volumes be opened at one time.
4. The specification of the standard forces DICOM applications to span through all seven layers of the protocol stack, as well as address application goals. This overspecification constrains freedom of developers when implementing such a system. This lack of

flexibility also prevents incorporating state-of-the-art technologies as they become available in the rest of a hospital network.

5. DICOM is based on a protocol stack that does not appear to be the worldwide mainstream protocol, although at one time the OSI protocol stack was expected (by some) to be the dominant one.

THE MEDICAL IMAGING INFORMATION MODEL (MIIM)

Description of the Model

The proposed MIIM is based on the DICOM information model and data dictionary. It is also a communications architecture that uses CORBA to ensure interoperability between different implementations of the model (examples of interoperability can be found on the Web⁸). Using CORBA allows a software component approach similar to the one suggested by Greenes and Bauman.⁹ MIIM provides a cleanly specified application programming interface (API) and a highly specified interface for input/output devices, specifically databases and storage systems. Having the operational characteristics of the interface specified actually provides flexibility and interoperability in the MIIM.

The idea of MIIM is to define explicitly and unambiguously the information that is accessible through the model and to describe the way to access that information, regardless of the computer or system language chosen to implement the model or application. To facilitate the programming of applications and implementations of the model, a set of libraries could be provided for the most common languages. In this way, the programmers are not

forced to learn CORBA programming, although full control of the communications should be possible if the programmer desires to do so.

The system can be divided into three parts:

1. *The object-oriented model* for the medical imaging information that the applications will use and access.
2. *The architecture* to allow the interchange between applications that use the information and providers of that information.
3. *The interface description* for the model describes how to access the information methods available for every object. This allows the programmers to create the applications without having to know how the model actually is implemented by every provider.

The Object-Oriented Model

The MIIM is based on the information model described in part 3, "Information Object Definitions," of the DICOM standard (shown in Fig 1). It gives an object-oriented point of view of the information necessary in the field of medical imaging. It describes the entities involved when dealing with medical images as objects. Data related to an entity can be accessed through methods in the corresponding object, and the relationships that exist between the entities also are maintained using object methods.

The MIIM model is a patient-oriented, intuitive representation of real-world hospital activities (shown in Fig 2). This means that the roots of our basic information system are the *patients*, upon whom we have performed various *studies*. To obtain a diagnosis for a study, we have produced

Fig 1. Basic information model.

Fig 2. Basic object-oriented model.

several *images* and *overlays*, which are grouped in *series* according to different criteria, eg, the image modality or equipment used to produce them.

This model could be extended to pathology, cardiology, and special studies, and can be considered a subset of the complete model that would comprise an RIS. The object-oriented description of the information system is independent of any implementation and is created to reflect the real world that the health care workers face when dealing with medical images.

MIIM Architecture

The architecture has been designed to guarantee the correct interchange of information between different implementations of the model explained above. To do that, it defines a set of minima that must be met by all products that want to claim conformance. This mandatory part of the architecture uses CORBA as the backbone to provide the interchange of information regardless of the language used to implement the server or the client application.

The architecture also allows the use of other ways of communication, if available. For example, Java remote method invocation (RMI) could be used when communicating from Java to Java, thus avoiding the conversion from Java objects to CORBA structures in the server, and from CORBA structures back to Java objects in the client. To

accomplish that, the architecture must provide additional standard, but optional, interfaces to cover these cases. Other ways of communicating could be possible, but they would be beyond the scope of this architecture, and therefore the intercommunication would not be guaranteed.

A series of libraries could be defined to help developers create implementations independent of CORBA, thus making their designs less sensitive to changes in the low-level specification. Libraries also would help programmers who were not familiar with CORBA programming because they could use the libraries for their favorite language and still be able to communicate with other implementations in other languages.

Figure 3 shows a possible scenario in which libraries have been created for Java and C++. The intercommunication is guaranteed because all the applications can be connected to any driver following the path through interface definition language and CORBA. However, an RMI optional path also is defined and allows Java Remote Applications to communicate directly with Java implementations that have this optional interface enabled.

The MIIM Interface Description

To guarantee the independence of the model from the implementation, there must be an unambiguous platform and language-independent interface to allow applications to interact regardless of

how the architecture works underneath, as well as to avoid confusion when implementing the model that could cause compatibility problems.

At the same time we chose CORBA as the core architecture, we chose the interface definition language as the way to express this interface because of its characteristics. The interface definition language is the basis for all implementations of the model, and it specifies the minimum set of requirements that all the standard-compliant products must satisfy.

With the interface definition language, we are able to ensure the desired cross-platform compatibility without forcing programmers to learn a new language, and we also use the standard libraries we defined to help developers with the architecture without requiring them to learn it. Each of these libraries defines a standard interface for a specific language, and because the libraries automatically translate that language to interface definition language, programmers can use them without losing compatibility. In the interface, the following must be specified clearly:

1. The object-oriented model for every specified language. This includes, at least, the interface definition language.
2. The interface to connect to a remote MIIM server, using the interface definition language or the standard libraries provided for the different languages specified.

3. The expected behavior of remote servers.

This part is language independent.

Servers can have different abilities depending on implementation, but *the model would guarantee only the ones defined in the standard*. It would be the developer's freedom and responsibility to use capabilities not included in the interface. However, different levels of servers could be defined in the standard interface, depending on their capabilities, to avoid creating a standard too limited for some providers or too demanding for others. The level of the server to which an application is connected could be queried in runtime, allowing the programmer to modify the behavior of the application depending on that parameter.

The interface allows us to divide the system into two separate points of view: the applications point of view and the service provider point of view.

The Applications Point of View

When the developers create a design, they can model the application using the definition of the objects provided by the interface, and they can rely on the behavior of these objects as it is specified. When the implementation time comes, they can use their language of choice and then use interface definition language to interact with the sources of information, or they can use the standard libraries directly, if available for that language, so they do

not need to be concerned about interface definition language.

From this perspective, the developer does not know and does not need to know how the model is implemented behind the MIIM interface. The application is completely detached from the model implementation. In fact, the choice between different implementations can be made even in runtime. Reliability and performance may be the basis for an implementation rather than using one because the application initially used it.

Moreover, the design and development of the application can focus on the application logic itself because the interface ensures the behavior of the defined object and hides the model implementation.

The developers also can decide the level of abstraction at which they want to work. They can program directly, use the specific libraries to control how the communication works, or use the communication-independent libraries that implement the model regardless of the method of communication. The interfaces of the standard provide all of this.

The last approach means that either interface definition language or another standard but optional way of communication can be used transparently, depending on availability (eg, in Fig 2, RMI could be used when connecting to RMI-enabled servers, and CORBA/interface definition language could be used otherwise, all with the same code).

Figure 4 shows the point of view of the application when dealing with the information system. The "DICOM Engine" object models the source

of the information so the application can be created without knowing how the model will be implemented.

The information can be accessed through the objects defined in the standard API, thus guaranteeing that the application will work with any standard-compliant implementation.

The Service Provider Point of View

From this perspective, the developers in charge of implementing the model can choose the appropriate elements to do so without having to worry about the applications that will use it. Because they do not have to learn details about image processing and how the images or the data will be used, they can concentrate on improving the performance and how to store and retrieve the data and images efficiently. This situation allows them to improve their products constantly, without fear of causing incompatibility problems with the applications.

The only concern for these developers is that their products must comply with the standard specification. The interface specification unambiguously defines the expected external behavior and allows developers to concentrate on the internals of their implementation of the model. The minimum behavior consists of something as simple as "Given a patient, return a list of all the studies that belong to that patient, in the specified format." The developers can implement the model as appropriate for them, as long as they respect the rules specified by the interface. This principle will be reflected, it is hoped, in the overall quality of the products

Fig 4. Application point of view.

because the customers will have the freedom to choose between different implementations without being constrained by the applications they already have. Therefore, they will be able to take into account more important issues, such as performance and reliability.

Along with the interface specification, a compliance test would be provided for two reasons: (1) developers would be able to check their implementations and certify that they adhere to the standard and (2) customers would have the warranty that their existing applications would work with the new implementations.

Figure 5 shows the perspective of the developers that must implement the model. There are only two important implementation concerns: (1) to follow the specification for the behavior of the DICOM engine and (2) to implement the model as efficiently as possible.

Advantages of MIIM

As well as fulfilling the previously mentioned goals, the design of the proposed model shows other interesting advantages:

- The object-oriented representation of a medical imaging information system makes applications easier to develop, hiding unnecessary complexity. This shortens the learning curve

for new developers in this field and provides numerous advantages.

- Detaching the sources of the information from the applications has two important consequences. First, from the developers' point of view, it allows them to concentrate on their main area of concern. The application developers can focus on the use of the information, without tying themselves to a specific implementation of the model. Information provider developers can put every effort into storage and retrieval of the information without being burdened with details about how the applications will use the data. From the customer's point of view, it creates an open market in which products can be chosen freely, without fear of being tied to a proprietary system.
- The existence of a recognized standard and a compliance test gives customers a warranty of compatibility and offers developers a tool to certify the quality of their products.
- Because the medical imaging field exists within the area of medical informatics, the object-oriented design, based in real-world activities, facilitates future integration with other information models based on the same premises. For example, the MIIM can be considered as a subset of the HIS and could be integrated easily inside an object-oriented representation of an HIS.

Fig 5. Storage point of view.

Limitations of MIIM

The design of the architecture and the way the model is created could raise some concerns about the suitability of the solution. In this section, we identify some of them, explain the reasoning behind some of our choices, and discuss how we address these concerns.

MIIM performance. Although it is true that a complete implementation in one language without using the different layers that the architecture provides could have better performance, other facts must be taken into account when considering the overall performance of the design:

- The main goal of the architecture is to ensure the interchange of information between different implementations in different platforms. It would be extremely difficult to accomplish the same with a complete one-language implementation without putting too many constraints on the developers.
- The architecture allows developers to choose the level at which they want to program their applications. It allows them to decide between using the standard libraries and taking advantage of a more general solution, or developing their solution by programming directly in interface definition language if more performance is needed. It also allows standard optional and more efficient methods of communication, like the RMI tunnel shown in Fig 3, which can be used if they are available to both sides.

If, despite all this, developers are not satisfied with the performance given by the model, nothing prevents them from creating a proprietary solution. This could be an option *if interchange of information with other implementations is not necessary* at that moment or in the future.

Compatibility with previous products. We must take into account the effort invested in previous products and standards and provide a bridge between the old systems and the new model. A series of tools and utilities should be provided to guarantee a smooth transition to the new model, and even though backward compatibility is not inherent to the proposed model, its design is based in the DICOM standard and such tools should not be difficult to implement.

Moreover, the characteristics and advantages of the new architecture provide enough incentives to

expect a steady, if not fast, migration to the object-oriented model.

The choice of a core architecture. The fact that CORBA/interface definition language was the choice for the core architecture to guarantee compatibility could raise some concerns about the future of a model being too tied to a particular technology. In the fast-moving field of distributed technologies, these worries are justified. However, MIIM is based in an object-oriented model that is independent of the technology. Therefore, even if we had to change the underlying architecture, the basic philosophy would be the same. On the other hand, if developers use the standard libraries provided for their favorite language, they need only to replace the old libraries with the ones that implement the new architecture. This flexibility allows applications to be independent of changes in a technology that is not their main focus, namely distributed architectures.

DISCUSSION

MIIM was designed using a DICOM engine that gives software engineers of imaging applications an interface to which they are accustomed. The engineers deal with an API and do not have to try to learn a novel object-like syntax. This allows the engineers to concentrate on their area of expertise. With MIIM, the combined talents of computer science specialists and image processing engineers can be brought to bear directly on problems in the field of medical imaging, which should benefit the health care industry by reducing integration efforts and software costs. The benefits of DICOM are used by MIIM. A DICOM engine is described, which must conform to the same interface as any other data input or output subsystem. Should DICOM remain the predominant mode of PACS intercommunication, the DICOM engine will change with the evolution of DICOM and continue to operate. However, MIIM allows more modern software interfaces to imaging objects in a true object-oriented fashion. It would allow a true implementation of the DICOM information model.

MIIM offers access to other I/O devices and systems via the lower interface layer. It is a more understandable interface to databases because CORBA has addressed these problems in the past. It can use JDBC, which has interfaces for a horizontal database market, not a specialized one. The interface of the MIIM requires database ven-

dors and storage device vendors to make their systems conform to the MIIM interface. Because the behavior of the interface is well defined, if a 3D holographic memory device becomes available, it is immediately usable, and upgrading from one long-term storage medium to the next will be trivial. Simply put, if a storage or database subsystem does not operate with MIIM, it is noncompliant. Those that conform to the MIIM interface are compliant and will interoperate with other MIIM devices.

A similar situation exists for applications using MIIM. They use MIIM having a well-understood and well-specified API with an expected behavior. This will make conformance statements unnecessary. If the software uses API, it conforms to MIIM and will work with other subsystems; otherwise it will not. This is because the lower-layer components already had to pass testing with the MIIM interface and the application software is using the API. This interface layer ensures that the API behaves as specified. Thus, a given set of software produces expected behavior provided by the interface.

The use of object-oriented programming will permit tighter integration of PACS, RIS, and HIS within an institution.⁹⁻¹⁰ The advantages of this approach extend beyond the walls of one hospital, as object-oriented programming and open architecture probably will prove to be the best way for handling enterprise systems that include several hospitals and outpatient facilities. It would also be important in regional health systems in the future.

The experience acquired while designing the architecture and developing the prototypes allowed

us to draw some preliminary conclusions from the use of MIIM:

1. One of the first things we could verify ourselves was the ease of use of the model and the improvements in development time. Once we had the model thought out, and even though we were creating the standard libraries concurrently, development of the prototypes was much easier and faster than expected. That the developers who worked on the project had no previous experience with DICOM and that most of them had no experience with the language chosen to implement the prototypes also were taken into account.
2. We realized that to guarantee the implementation independence of the model and to provide the standard libraries, an independent institution should be created or adapted to coordinate the definition and put the standard into practice. This institution would be in charge of defining the standard interfaces for the languages specified and for developing the standard libraries to facilitate the use of the model. Because the interfaces and libraries can detach the applications from the implementation of the architecture, the entire industry would benefit when improvements in the libraries take place.
3. Developing the prototypes and the first set of libraries tested some of the advantages that MIIM provides. The language chosen for this first implementation was Java. There was both the standard implementation and the optional all-Java implementation. Both implementations worked well.

REFERENCES

1. National Electrical Manufacturer's Association: Digital imaging and communications in medicine (DICOM), 1996 (NEMA Standards Publication PS3.1 through PS3.13).
2. Object Management Group: A discussion of the object management architecture, January 1997
3. Object Management Group: CORBA services; Common Object Services Specification, November 1996
4. "CORBA and OMG information resources," <http://www.acl.lanl.gov/CORBA/>
5. University Hospital of Geneva, PAPYRUS: Image file format specifications, version 2.3, 1991
6. ACR-NEMA Standards Publication no. 300-1988, Digital image and communications, National Electrical Manufacturers Association, Washington, DC, 1989
7. Rose MT: Transition and coexistence strategies for TCP/IP to OSI. *IEEE J Selected Areas Communications* 8:57-66, 1990
8. CORBAnet: The ORB Interoperability Showcase, <http://www.corba.net/>
9. Greenes RA, Bauman RA: The era of health care reform and the information superhighway. *Radiol Clin North Am* 34:463-468, 1996
10. Chimiak WJ, Lopez D, Martinez R: An architecture for hospital information integration. *Proceedings of the SPIE Medical Imaging Conference 1999, PACS Design and Evaluation: Engineering and Clinical Issues*, February 1999, pp 10-15