TECHNOLOGY AREA

Last Updated: 03-09-05

DEFINITION	
Name	Distributed Object Interoperability
Description	Distributed object technology allows objects on different machines to communicate messages to each other. A distributed object is a piece of code that can live anywhere on a network. They are packaged as independent pieces of code, which can be accessed by remote clients via method invocations. The language and compiler that are used to create distributed objects are transparent to their clients. Clients do not have to know where on the network, a distributed object resides, or what machine it is on, whether it is the same machine or another. It can be executed on a totally different operating system. Distributed objects are therefore able to message each other anywhere in the world.
Rationale	Given the growth of client/server computing which has spread from single server departmental Local Area Networks (LAN) to Wide Area Networks (WAN), there needs to be a method of developing applications that exist on multiple servers in disparate networks rather than a single stand alone server. Distributed objects allow for this distribution of computing resources.
Benefits	• Ease of programming: the underlying object model, the Interface Definition Language (IDL), and the supporting tools combine to simplify the task of writing distributed applications. • Extensibility and manageability: systems built from interacting objects are inherently extensible since objects can be easily added or replaced. Extensibility allows customized management interfaces to be added to the system. • Encapsulation and systems integration: distributed object technology has proved very effective in encapsulating "legacy systems" as objects. These objects offer interfaces that can then be used to implement an integrated system. A server built from distributed objects has several advantages over existing servers: • It is easily and transparently distributed. If the underlying object system supports migration, then data can move to be near the point of access automatically; similarly, replication for reliability and/or availability can be done transparently. • It is extensible. As new applications arrive, objects to handle them can be added to the implementation as easily as CGI programs may be added to current servers, but with the benefits of strong type safety and better configuration management than is currently possible. • Load balancing can be implemented, so that frequently-accessed objects remain 'live', while other objects may be cached to disc between invocations; live objects can be spread across a set of machines according to demand. The use of distributed object technology makes it easy to develop long-life client-server applications through clearly defined server and client APIs.

ASSOCIATED ARCHITECTURE LEVELS	
Specify the Domain Name	Interoperability
Specify the Discipline Name	Application Interoperability

KEYWORDS			
List Keywords	Distributed objects interoperability, objects		
ASSOCIATED COMPLIANCE COMPONENTS			
List the Compliance Component Names	Distributed Object Interoperability Guidelines Web Services		
ASSOCIATED PRODUCT COMPONENTS			
List the Product Component Names	SOAP DCOM/COM EJB UDDI CORBA		
CURRENT STATUS			
Provide the Current Status	☐ *In Development* ☐ *Under Review* ☒ *Approved* ☐ *Rejected*		
AUDIT TRAIL			
Creation Date	03/09/2005	*Date Approved / Rejected*	10/11/05
Reason for Rejection			
Last Date Reviewed		*Last Date Updated*	
Reason for Update			