De otro lado, en la investigación en inteligencia artificial (I.A.), y específicamente para el problema conocido como *representación del conocimiento*, Marvin Minsky propuso el concepto de *marco* (ingl. "frame") [Minsky, 75] como una estructura flexible y poderosa, y -a su manera- también apuntaba a la orientación por objetos.

Los *marcos* resultaron ser muy similares a los *objetos*, aunque esa similitud se nubla por la utilización de otros nombres para conceptos a veces iguales: la *especialización de ejemplares* específicos a partir de marcos más genéricos corresponde a la *herencia*, los marcos también se organizan jerárquicamente, las *ranuras* ("slots") de los marcos corresponden a los *atributos* de los objetos, los *procedimientos ligados* ("attached procedures") de los marcos corresponden aproximadamente a los *métodos* de los objetos y, por último, los marcos usan *reglas de producción* mientras los objetos envían *mensajes*.

Desde el punto de vista de la I. A., actualmente los sistemas basados en marcos y los O.O. convergen hacia formas poderosas de representación del conocimiento en las cuales no tiene mucho sentido distinguir entre los dos orígenes.

Es posible relacionar el enfoque O.O. en el dominio de la ingeniería de software con otras tendencias de enfoque que se dan actualmente en otros dominios, caracterizadas por modelos con *objetos* o *agentes* interactuantes, por ejemplo:

En las *teorías sobre la mente*, los modelos *monolíticos*, en los que se suponía una entidad global, con una unicidad característica, van dando paso a modelos con múltiples componentes, como la *Multimente* [Ornstein, 86], y la *Sociedad de la Mente* [Minsky, 86]. Estos modelos tratan de explicar, respectivamente, la incoherencia de nuestros comportamientos y el surgimiento de comportamientos inteligentes.

En *robótica*, también el enfoque monolítico va cediendo ante la arquitectura de *subsumisión* [Brooks, 86] en la cual el comportamiento final surge a partir del juego de varios agentes de comportamiento elementales.

En las *redes globales* como Internet, la tendencia para mejorar el acceso a la información disponible es tener *agentes de usuario*: entidades de software que "exploran" la red recolectando información que satisfaga los intereses de cada usuario y le informan periódicamente sobre lo hallado.

En *inteligencia artificial*, además de los *marcos* ya mencionados, las arquitecturas con *tablero* o pizarrón (ingl. "blackboard"), así como la *I. A. Distribuida* también utilizan el modelo de agentes interactuantes para resolver problemas complejos.

En fin, hasta los mismos *virus informáticos*, aunque indeseables por sus efectos, no dejan de ser en sí mismos entidades interesantes como ejemplo de agentes que despliegan cierta autonomía.

Parecería pues, dados los cinco ejemplos anteriores, que el enfoque por objetos o agentes ha surgido también en el mundo de la ingeniería de software como metodología correcta para tratar con la complejidad y que las ventajas que ofrece no dejan duda respecto a que se deben acoger.

Se reconoce que algunos lenguajes como SIMULA, Smalltalk y Eiffel llevan el concepto de O.O. de una manera innata, pura y consistente, mientras que otros lenguajes tradicionales han sido *adaptados* o *extendidos* a este enfoque objetual.

Lo aceptable de la adaptación depende de la flexibilidad del lenguaje base para crearle las nuevas estructuras exigidas por la orientación por objetos, por ejemplo el LISP es sumamente flexible, llegándose a considerar no sólo como un lenguaje de programación, sino como un "lenguaje de programación *programable*" [Foderaro, 91]. Esta característica permite en este lenguaje incorporar e inclusive sobrepasar lo propuesto por nuevos paradigmas. Para la programación O.O. se han creado tres sistemas en este lenguaje: Flavors, Loops y más recientemente CLOS.

En el caso de Pascal, la orientación por objetos puede comenzar por incorporar una nueva