

# SIO = Shared Integrated Ontologies (One part of the OOR)

COLORE: CL Ontology modules and their relations  
(Gruninger)

PIFO = Primitive Inventory Foundation Ontology (suggested  
basis for semantic interoperability)

COSMO = Common Semantic Model (Cassidy)

FO = Foundation Ontology (any ontology including the most  
basic concepts, used as a foundation for constructing  
domain ontologies – e.g. CYC 'BaseKB', SUMO)

# Principle of the PIFO

- Accurate semantic interoperability requires agreement on a common Foundation Ontology, or a complete set of mappings and translations among the interoperating ontology systems.
- If a Foundation Ontology contains all of the identified semantic primitives (i.e. is a PIFO), it will be able to logically specify the meanings of any elements in domain ontologies that use it as a FO
- Creating the elements of a domain ontology as combinations of elements in the PIFO automatically maps those ontologies, and enables automatic merging and translation of assertions from one domain ontology form to another.

# COSMO

- COSMO is an ontology being developed as a first approximation to a PIFO. It will contain representations of all of the words (2148 of them) in the Longman Dictionary defining vocabulary – words sufficient to linguistically define all of the other words in the dictionary. This is the best current inventory of terms likely to represent most of the needed primitives. Elements other than primitives will be included for convenience.
- Currently in the OWL format, COSMO needs to be converted to some FOL version to function as a PIFO. There are still about 150 Longman words not yet represented.

# COSMO in the SIO

- The ontologies in the SIO will ideally be integrated with each other, so that the relations between them are clear.
- This is a very time-consuming task, and integration is likely to be incomplete for some time.
- If any domain ontologies are created using the COSMO as a PIFO, those would be integrated directly with the COSMO.
- The CL version of COSMO may use or map to modules from COLORE.