

## **Актуальные потребности применений интероперабельных систем**

В начале рассматриваются насущные потребности применений, определяющие существенную мотивацию для перехода к интероперабельным информационным системам и разработки соответствующей технологии.

**1. Функционирование систем в условиях информационной и реализационной неоднородности, распределенности и автономности информационных ресурсов системы.** Информационная неоднородность ресурсов заключается в разнообразии их прикладных контекстов (используемых онтологических средств - понятий, словарей; отображаемых реальных объектов, составляющих "поверхность соприкосновения" различных реальных миров и их (объектов) абстракций в информационных системах; семантических правил, определяющих адекватность совокупностей моделируемых объектов реальности; моделируемых деятельности; видов данных, способов их сбора и обработки; интерфейсов пользователей и т.д.).

Реализационная неоднородность источников проявляется в использовании разнообразных компьютерных платформ, средств управления базами данных, моделей данных и знаний, средств программирования, операционных систем и т.п.

**2. Интеграция систем.** Системы эволюционируют от простых, автономных подсистем к более сложным, интегрированным системам, основанным на интероперабельном взаимодействии компонентов.

**3. Реинжиниринг систем.** Эволюция деловых процессов - это непрерывный процесс, который является неотъемлемой составляющей деятельности организаций. Соответственно, создание системы и ее реконструкция (реинженерия) - непрерывный процесс формирования, уточнения требований и конструирования. Реконструкция систем осуществляется постепенно. Система должна быть сконструирована так, чтобы произвольные ее составляющие могли быть реконструированы при сохранении целостности системы.

**4. Миграция унаследованных систем.** Любая система после создания противодействует изменениям и имеет тенденцию быстрого превращения в бремя организации (т. н. legacy systems - унаследованные системы, использующие "уставшие" технологии, архитектуры, платформы, а также собственно программное и информационное обеспечение, при проектировании которых не были предусмотрены нужные меры для их пошаговой миграции в новые системы, соответствующие новым требованиям деловых процессов и технологии). Существенно, что в процессе миграции необходимо, чтобы мигрировавшие составляющие системы и оставшиеся компоненты унаследованных систем сохраняли интероперабельность.

**5. Повторное использование неоднородных информационных ресурсов.**

Технология разработки информационных систем должна позволять крупномасштабно применять технологию повторного использования информационных ресурсов, переходя от технологии программирования, основанной на интенсивном индивидуальном труде по созданию вручную изделий, удовлетворяющих специфическим требованиям одного конкретного применения, к технологии, основанной на планируемых капиталовложениях в разработку повторно-используемых компонентов, которые могут быть "соединены" (т.е., образованы их интероперабельные сообщества) для производства серий стандартизованных продуктов в определенной прикладной области.

**6. Продление жизненного цикла систем.** В условиях исключительно быстрого технологического развития требуются специальные меры, обеспечивающие необходимую продолжительность жизненного цикла.

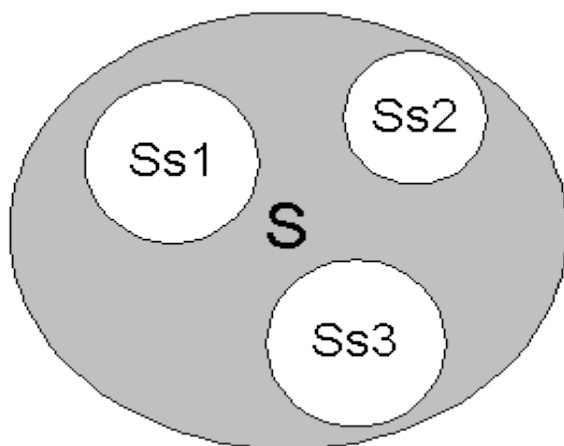
Существенно, что свойство интероперабельности информационных ресурсов является необходимой предпосылкой удовлетворения перечисленных требований.

### **Эволюция роста системы с точки зрения взаимодействия ее подсистем**

Попробуем описать возможную эволюцию роста некоей гипотетической ИС.

#### **Этап 1**

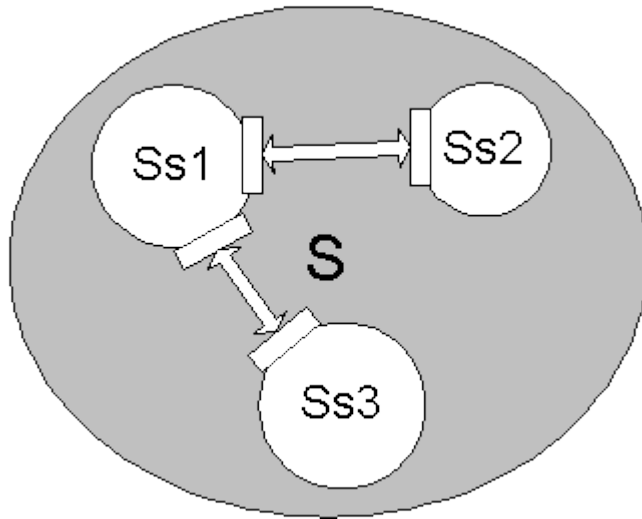
Допустим, когда-то была создана система S (см рис.). Эта система состоит из 3-х независимых между собой подсистем (Ss – Subsystem).



#### **Этап 2**

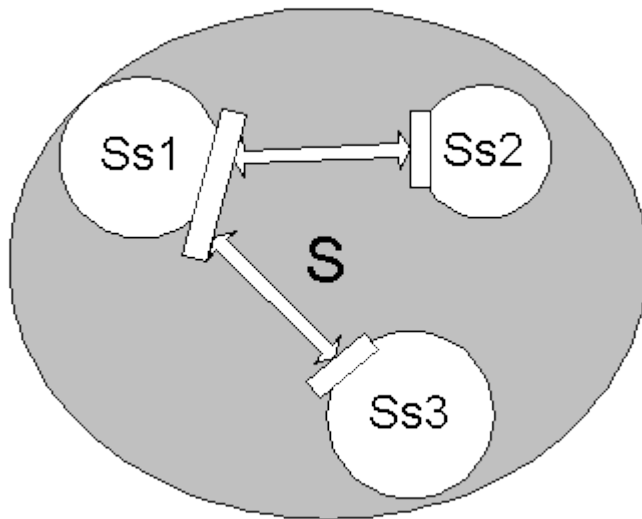
На очередном этапе развития системы возникла потребность взаимодействия между Ss1 и Ss2, Ss1 и Ss3.

Разработчик пишет интерфейсы для каждой связи  $Ss-i \leftrightarrow Ss-j$ . Обозначим стрелками направления взаимодействия между подсистемами. Интерфейсы представляют из себя описания вызовов процедур, исполняемых на данной подсистеме. Стрелки - это передача параметров вызова и возвращаемый результат.



### Этап 3

Далее, разработчик видит, что интерфейсы к  $Ss1$  содержат пересекающееся множество описаний функций, и он решает объединить их в один интерфейс.

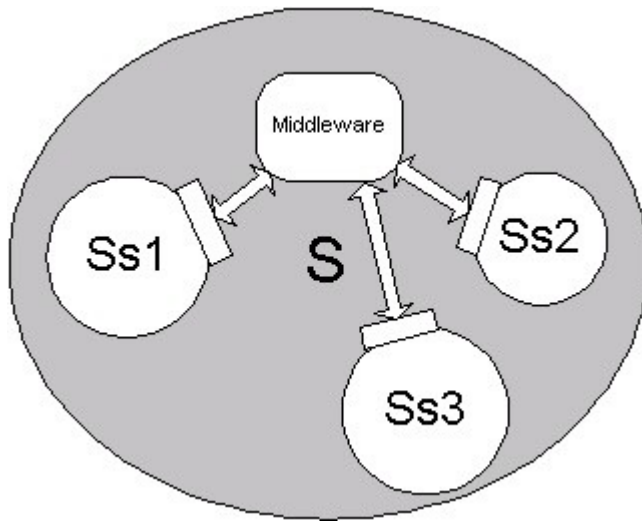


### Этап 4

Предположим, что система усложнилась до такой степени, что стало слишком накладно каждый раз писать индивидуальные интерфейсы для новой подсистемы

(вернее, ее части, которая обращается к остальным Ss).

Возникает идея создания промежуточного программного слоя-шины, который инкапсулировал бы в себе эту задачу. Назовем его "Middleware" (связующее ПО).



## Этап 5

На сегодняшний день уже разработано несколько стандартов такого промежуточного ПО, среди них OMG CORBA и Microsoft DCOM/COM+. Поэтому в какой-то момент рациональнее будет не изобретать велосипед, а использовать эти стандарты для построения их собственной реализации либо продукты сторонних производителей ПО. Это прямой путь к технической интероперабельности подсистем.