**4. Внешний интерфейс CLI**

Common Language Infrastructure

Разработан Microsoft. Менее безопасный чем JNI. В C# и C++ есть «небезопасное» множество размером примерно с С

* Прямое управление раскладкой полей объекта
* Уменьшает гибкость системы
* Закрепление объекта в памяти и получение указателя
* Уменьшает гибкость менеджера памяти
* Потенциальная угроза безопасности
* Прямая работа с указателями из управляемого кода
* Потенциальная угроза безопасности
* pinvoke (.NET)
* Связывание произвольной внешней функции
* копирование объектов с изменением раскладки полей

Функции, реализованные в неуправляемых библиотеках DLL, могут вызываться из управляемого кода с помощью вызова неуправляемого кода P/Invoke. Если исходный код библиотеки DLL недоступен, вызов P/Invoke является единственным вариантом обеспечения взаимодействия. В отличие от других языков .NET в Visual C++ предусматривается альтернатива вызову P/Invoke.

Все совместимые языки компилируются в Common Intermediate Language(CIL), который является промежуточным языком, абстрагированным от платформенного «железа». Во время исполнения кода CIL будет перекомпилирован в машинный язык, соответствующий конкретному «железу» и ОС.

Промежуточное представление(IR) :

* Язык регистровой машины
  + На 45% меньше операций доступа к памяти
  + Возможность оптимизации на уровне IR
  + Используется в статических и оптимизирующих JIT компиляторах
* Язык стековой машины
  + На 25% более компактен
  + Сохраняет семантику исходного кода(проще восстановить исходный код)
  + Проще для интерпретации и шаблонной компиляции
* Используется в JBC(Java Beans?) и CIL

Оба языка полны по Тьюрингу(эквивалентны машине Тьюринга)