Проект Inferno, родился в недрах Lucent Technologies. Inferno представляет собой целостную среду, по самой своей конструкции идеально приспособленную для распределенных сетевых вычислений благодаря компактности, масштабируемости и способности становиться органическим элементом других операционных систем.

Это портируемая операционная система, созданная для построения распределённых и сетевых систем на широком диапазоне устройств и платформ. Эта система обладает межплатформенной переносимостью и может выполняться как пользовательское приложение или как независимая операционная система. Поддерживается для большинства широко распространенных операционных систем и платформ. Каждая система Inferno предоставляет пользователю идентичную среду разработки независимо от основной операционной системы или архитектуры, разрешая работать в гомогенной среде с множеством различных платформ.

Существует минимум три задачи, с которыми Inferno справится лучше других.

* Распределённые вычисления и Grid (благодаря протоколу Styx доступ к локальным и удалённым ресурсам ничем не отличается, что позволяет очень просто писать распределённые системы).
* Гомогенная среда (приложение написанное на Limbo будет гарантированно одинаково работать и под виндой, и под юниксом, и на встроенных устройствах).
* Встроенные устройства (Inferno поддерживает множество процессоров и устройств… возможно Inferno уже работает внутри Вашего гаджета, просто вы об этом не подозреваете

В основу Inferno положены три простых принципа. Все они связаны с сетью и "сетецентрической" моделью.

* Первый состоит в том, что обращение ко всем ресурсам, доступным прикладной программе, осущесствляется на основе единого протокола. Этот протокол, получивший название Styx, обеспечивает взаимодействие и с локальными ресурсами, такими как локальные файлы и диски, и с удаленными, такими как файлы на других машинах и серверы баз данных.
* Второй принцип - это представление всех ресурсов в виде файлов в иерархической файловой системе (так же, как в Unix). Операции открытия, чтения, записи и закрытия применяются к принтерам, базам данных, сетевым соединениям, удаленным файлам. То, что все ресурсы рассматриваются как файлы, значительно упрощает организацию сети с точки зрения прикладной программы.
* Третий принцип, на котором основана Inferno, заключается в том, что прикладные программы работают с сетью и только с сетью. Элементы файловой системы могут представлять собой смесь локальных и удаленных ресурсов, но для прикладных программ это не имеет значения. Обращения к ресурсам, разбросанным по разным машинам сети, ничем не отличаются для них от обращений к локальным ресурсам своей собственной машины.

Система Inferno имеет полный набор компонентов, необходимых для ее установки непосредственно на. Такой вариант называется установкой в естественной среде (native environment).

Можно установить Inferno и поверх уже имеющейся на машине ОС, например Unix или Windows NT, осуществив так называемую установку в искусственной среде (emulation environment). Эта статья относится в основном к функционированию Inferno в искусственной среде - поверх Windows NT 4.0. Экземпляр Inferno можно получить по адресу

ОС Inferno состоит из четырех отдельных частей[**\***](https://www.osp.ru/data/www2/pcworld/1997/10/54.htm#footnote):

* Операционная система Inferno (ядро).
* Язык программирования Limbo.
* Виртуальная машина Dis.
* Протокол связи Styx.

Язык программирования Limbo и виртуальная машина Dis очень схожи соответственно с Java и Java-машиной. Limbo - это язык, синтаксически близкий к Си (а не к Си++, как Java); он предназначен для написания прикладных программ, работающих под управлением Inferno. Компилятор Limbo генерирует не зависящий от системы байтовый код, который может исполняться на любой виртуальной Dis-машине. Таким образом, программы, написанные на Limbo, будут выполняться на любой машине, поддерживающей Inferno. В настоящее время Limbo - единственный язык программирования для Inferno.

Сердце системы - это ее ядро (собственно Inferno) и протокол Styx. Ядро отвечает за низкоуровневые задания, такие как распределение ресурсов, защиту и взаимодействие с аппаратурой. Сетевые операции обеспечиваются протоколом Styx, который, в свою очередь, подчинен ядру. В силу очевидных причин определенные части ядра зависят от аппаратуры, так что для разных машин и конфигураций необходимы разные ядра. Код ядра написан на стандартном Си (ANSI C), так что его относительно просто перенести в любую вычислительную среду, для которой существует компилятор языка Си.

Ядро

**Ядро** - нижний уровень программного обеспечения Inferno, поддерживающий функционирование уровней сети и прикладных программ. Ядро отвечает за управление процессами, памятью, вводом-выводом и пространством имен (name space).

Память организована в виде несбалансированного B-дерева, листы которого рассортированы по размеру. Когда процесс запрашивает память, система просматривает B-дерево и находит в нем блок, удовлетворяющий запросу. Если размер блока более чем на 25% превышает требуемый, он разбивается на две части, одна из которых отдается процессу, а другая присоединяется обратно к B-дереву. Если разница в размере меньше 25%, блок отдается процессу целиком и оставшееся пространство не используется.

Другая задача, которую естественно решать ядру ОС, - это распределение времени между процессами. В Inferno различаются восемь классов (приоритетов) процессов, причем каждому соответствует отдельная очередь. Inferno выделяет время фиксированными порциями и обслуживает все восемь очередей по круговой схеме.

Все устройства, как и в Unix, включены в файловую систему. Символ "/" обозначает корневую файловую систему. Имена остальных устройств представляют собой последовательности из символа "#" ("решетки") и некоторой буквы. Например, #V означает телевизионные устройства, а #D - уровень защиты гнезд. Связь с устройствами осуществляется по сети.

**Styx**

Протокол связи Styx используется устройствами Inferno для взаимодействия с сетью и совместного использования сетевых ресурсов. Сетевые соединения, естественно, представлены как часть файловой системы. Файловая система сервера обычно содержит представления тех сетевых служб, которые обеспечиваются данным сервером, а также тех, которые ему доступны. Чтобы клиент мог получить доступ к ресурсам, на нем должно быть создано то же их представление, что на сервере. Это осуществляется путем монтирования и подключения.

В Styx различаются два типа сообщений: передача начала протокола (T-сообщение) и передача ответа (R-сообщение). Указание на тип содержится в первом байте сообщения. Следующие два байта представляют собой метку (tag), устанавливаемую клиентом и служащую для идентификации сообщения. Каждое исходящее от клиента сообщение (за исключением абсолютно тривиальных) должно иметь уникальную метку, позволяющую его однозначно идентифицировать. Четвертый и пятый байты - это идентификатор файла, над которым производится операция; они имеют смысл только в том случае, когда сообщение относится к файлу. Оставшаяся часть сообщения содержит данные, определяемые выполняемой операцией.

По своему назначению сообщения Styx подразделяются на четыре категории:

* перемещение в иерархии файлов;
* доступ к данным, содержащимся в файлах;
* управление атрибутами файлов;
* разное (синхронизация, сообщения об ошибках).

**DIS**

Программы на Limbo интерпретируются виртуальной машиной Dis. Она отвечает за распределение памяти и времени, а также за "сборку мусора". На компьютере всегда работает только один экземпляр Dis, на который бывает возложено управление кучей: он извлекает из кучи и выдает программе необходимую ей память.

При управлении памятью Dis опирается на описатели типа (type descriptors). Каждому объекту, занимающему память, приписан описатель типа, содержащий размер объекта и указатели на другие объекты в куче. Соответствующая информация генерируется компилятором Limbo и используется Dis-машиной для обеспечения защиты памяти.

Потоки, взаимодействующие с другими нитями, могут, кроме того, приостанавливать работу. Если поток переведен в состояние send (посылка сообщения), она останется в этом состоянии до тех пор, пока поток-адресат не подтвердит, что сообщение получено. Аналогично ведут себя нити и при входе в состояние receive (получение сообщения).

В определенных случаях ресурс бывает нужен ядру. Тогда оно блокирует нить, использующую этот ресурс, на время выполнения жизненно важных системных функций.

В Dis используется два типа механизмов "сборки мусора". Первый - это счет ссылок (reference counting), при котором использованная память становится доступной сразу после освобождения. Второй - маркирование и очистка (mark-and-sweep). Он применяется только к циклическим структурам и выполняется как отдельный фоновый процесс.

**Limbo**

На Limbo пишутся программы для Inferno. Синтаксис Limbo очень похож на синтаксис Си (но не Си++). Язык поддерживает основные типы данных, циклы и структуры принятия решений, а также надежный коммуникационный интерфейс. Программа на Limbo состоит из единиц, называемых модулями. Каждый модуль содержит интерфейсную декларативную секцию (declaration part) и секцию реализации (implementation part). В декларативной секции перечисляются функции и данные, предоставляемые модулем, а также типы данных и всевозможные константы. В секции реализации находятся реализации функций модуля и, если нужно, реализации вспомогательных внутренних функций.

Поскольку Limbo основан не на Си++, а на Си, он не является объектно-ориентированным в том же смысле, что Java. На настоящий момент Limbo - единственный язык, поддерживаемый в Inferno