Мобильность программного обеспечения

Мобильность программного обеспечения - способность программного обеспечения работать на различных аппаратных платформах или под управлением различных операционных систем.

Мобильность программного обеспечения

Проблема мобильности программного обеспечения возникла с появлением первых ЭВМ с гибким программным управлением (начало 1950-х).

Основоположником информатики и автоматизации программирования, является Алексей Андреевич Ляпунов, введший понятие ***программы*** как последовательность чередующихся этапов, на которых выполняется некая обработка данных. Этап Ляпунов предложил назвать оператором, а схемой счета — совокупность операторов и логических условий. Схема и совокупность спецификаций каждого оператора — это программа. Такой взгляд в 50-х был революционным и сразу стал основой первых трансляторов (компиляторов) или ***программирующих программ***, как их тогда называли.

Мобильность программного обеспечения

Проблема мобильности программного обеспечения возникла с появлением первых ЭВМ с гибким программным управлением (начало 1950-х).

Первым в мире транслятором языка высокого уровня является ПП (Программирующая Программа), он же ПП-1, созданный молодыми математиками С. С. Камыниным и Э. З. Любимским в 1954 году, основе Операторного метода А.А.Ляпунова.

Транслятор ПП-2 (1955 г., уже 4-й в мире транслятор) уже был оптимизирующим и содержал собственный загрузчик и отладчик, библиотеку стандартных процедур (по признанию Эдгара Кодда).

Мобильность программного обеспечения

К середине 1960-х годов программисты осознали, что компиляторы с одного и того же языка для разных компьютеров содержат много общего, и появилось желание научиться создавать такие трансляторы, которые могли бы функционировать на различных компьютерах.

В США попытались создать универсальный компилятор, изготовляющий программы для разных компьютеров на основе их формализованного технического описания (UNCOL). Однако, такого описания найти не удалось, - его и до сих пор никто не придумал.

Мобильность программного обеспечения

Концепция программной совместимости впервые в широких масштабах была применена разработчиками системы IBM/360. Основная задача при проектировании всего ряда моделей этой системы заключалась в создании такой архитектуры, которая была бы одинаковой с точки зрения пользователя для всех моделей системы независимо от цены и производительности каждой из них.

Огромные преимущества такого подхода, позволяющего сохранять существующий задел программного обеспечения при переходе на новые (как правило, более производительные) модели были быстро оценены как производителями компьютеров, так и пользователями и начиная с этого времени практически все фирмы-поставщики компьютерного оборудования взяли на вооружение эти принципы, поставляя серии совместимых компьютеров.

Следует заметить однако, что со временем даже самая передовая архитектура неизбежно устаревает и возникает потребность внесения радикальных изменений архитектуру и способы организации вычислительных систем.

Мобильность программного обеспечения

Мобильность программного обеспечения

Болезнь второй системы

Требовать и эффективности, и гибкости от одной и той же программы - все равно, что искать очаровательную и скромную жену... по-видимому, нам следует остановиться на чем-то одном из двух.

Фредерик Брукс-младший

Фредерик Филлипс Брукс — младший

Мобильность программного обеспечения

Мобильность программного обеспечения - способность программного обеспечения работать на различных аппаратных платформах или под управлением различных операционных систем.

1. Семейства ЭВМ
2. Перенос текстов на ЯВУ и Макропроцессоры
3. Мобильные ОС
4. Универсальные программные интерфейсы (УПИ)

Мобильность программного обеспечения

В втором методе мобильность достигается за счет переноса исходных текстов ПО с инструментальной ЭВМ на объектную, на которой производится их перетрансляция и дальнейшая обработка объектных модулей (создание библиотек, компоновка и т. д.).

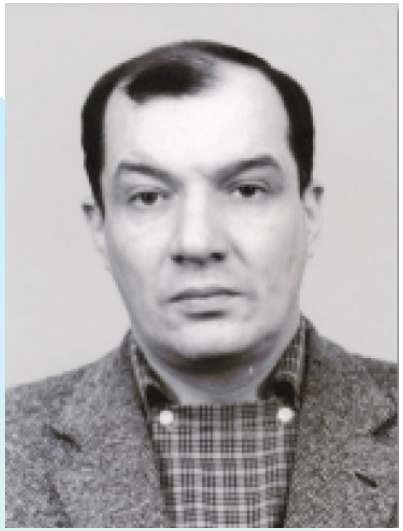
Метод требует наличия компилятора с используемого ЯВУ как на инструментальной, так и на объектной ЭВМ, причем реализации ЯВУ на обеих ЭВМ должны быть достаточно близки.

С ростом объема ПО сложность проблем, связанных с применением данного метода, растет.

Мобильность программного обеспечения

Постановка мобильных ОС решает многие вопросы по переносу ПО, но, во-первых, не всегда возможна, так как требует определенной поддержки со стороны аппаратного обеспечения, а во-вторых, заставляет пользователя работать в жестко заданной программной среде, не всегда адекватной его проблемной области.

Мобильность программного обеспечения



В 1963 году С.С. Камынин и Э.З. Любимский вместе с В.В.Луциковичем предложили совершенно другой подход к решению проблемы мобильности - не технический, а алгоритмический. Для создания универсальных (мобильных) компиляторов, способных функционировать на компьютерах различных типов, ими был разработан специальный алгоритмический машинно-ориентированный язык абстрактной машины АЛМО.

Мобильность программного обеспечения АЛМО

Одним из способов решения проблемы программной совместимости электронных вычислительных машин является применение некоторого общего языка программирования и трансляторов с этого языка.

Эта особенность машины АЛМО обеспечивает универсальную сферу применения языка АЛМО, так как позволяет для любого класса задач и любой из этих машин получить «опосредованную» программу, имеющую почти такую же эффективность, как и программа, написанная для конкретной машины.

Мобильность программного обеспечения АЛМО

Он одновременно использовался и как язык для написания компиляторов, и как язык для оформления составляемых этими компиляторами программ.

Для каждого конкретного типа компьютера оставалось разработать только один компилятор с языка АЛМО, а все остальное «раскручивалось» автоматически.

По своему назначению и используемым средствам АЛМО явился прототипом созданного значительно позднее популярного сейчас языка «С».

Мобильность программного обеспечения АЛМО

Язык системного программирования (машинно-ориентированный язык), задумывался как язык-посредник при трансляции с различных языков. Идея состояла в том, что для каждой аппаратной платформы достаточно было написать транслятор Алмо — и ты уже можешь работать с множеством языков программирования, которые имели трансляцию в Алмо. Были созданы реализации языка для основных отечественных машин того времени (М-20, БЭСМ-6, Минск 2, Урал 11, ЕС ЭВМ) и трансляторы с Алгола-60 и ФОРТРАНа, Паскаля в Алмо, причем все трансляторы также были написаны на Алмо и “раскручены” на всех этих машинах.

Адекватные языки нужны были не только для трансляторов, но и для всей возникающей области системного программирования: начали появляться и другие языковые процессоры, первые операционные системы, информационные системы.

В связи с этим и у нас, и на Западе начали появляться специальные языки, предназначенные для системного программирования.

Характерной особенностью первого поколения этих языков (связанной с необходимостью хорошо учитывать архитектуру и машинное представление данных) была машинная ориентированность. Разрабатывались эти языки, как правило, в коллективах, имевших большой опыт в создании системных программ - таковыми в тот период были системы программирования.

Отечественные языки Алмо, Эпсилон, Сигма, которые были одними из первых в мире языками системного программирования, создавались поэтому в коллективах трансляторщиков, только что завершивших большие программные проекты и почувствовавших, каково сапожнику обходиться без сапог.

UNIX-BESYS

Мобильность программного обеспечения

Корни UNIX растут из середины 50-х годов (57-58) прошлого века, когда были предприняты первые попытки обеспечить разделение доступа к компьютерам.
Под руководством Виктора Высотского была создана операционная система BESYS.

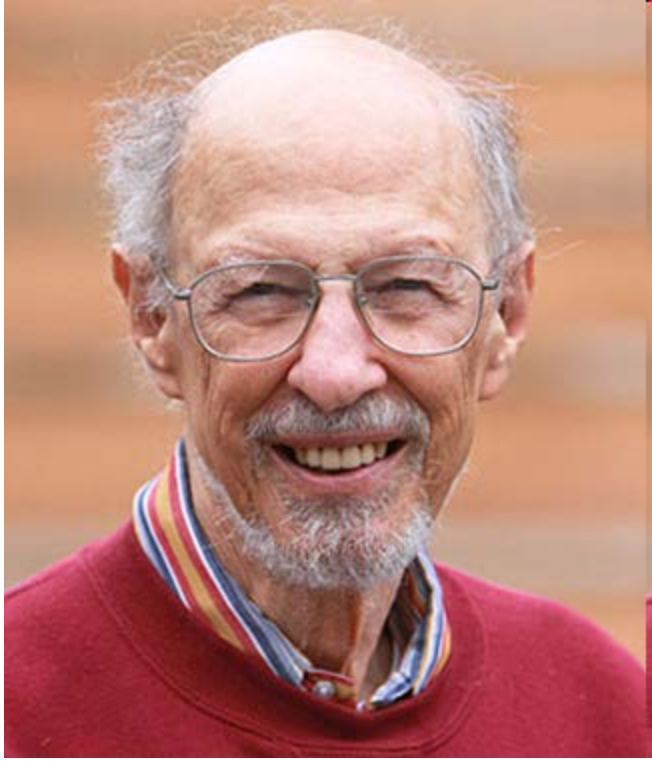
Система BESYS (Bell Operating System) базировалась на компьютерах IBM 709 и IBM 7094, к которым было присоединено дополнительное оборудование для скоростной обработки перфокарт (через компьютер IBM 1401), и печати результатов на бумаге.

Мобильность программного обеспечения

Мобильность программного обеспечения

UNIX-MULTICS

Система Multics (1964) имела множество характерных особенностей, обеспечивавших её безотказность и высокую производительность. Модульность программного обеспечения Модульность электронных устройств, что позволило наращивать вычислительные возможности системы простой заменой её модулей: центрального процессора, памяти, дискового пространства, и т. д. Отдельные для каждого пользователя списки доступа к файлам обеспечили весьма гибкий механизм коллективного использования информации в системе, гарантирующей также обеспечение полной конфиденциальности хранимой и используемой пользователями информации.

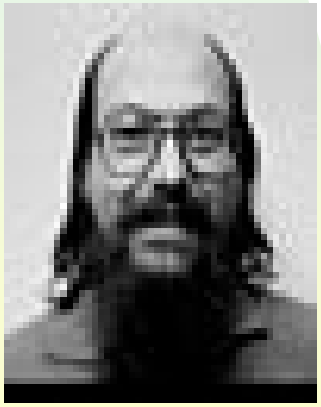


Fernando José Corbató

UNIX

В 1968 Сотрудники фирмы Bell Laboratories Кен Томпсон и Деннис Ритчи приступили к разработке операционной системы ***UNIX.***

Dennis M. Ritchie



Ken L. Thompson

UNIX

В 1972 году Bell Laboratories начала выпускать официальные версии ***UNIX***.

Президент Клинтон вручает национальные медали в области технологии
Кену Томпсону и Деннису Ритчи

27.04.1999, Вашингтон

UNIX

UNIX

UNIX

- многопользовательский режим со средствами защиты данных от несанкционированного доступа, реализация мультипрограммной обработки в режиме разделения времени, основанная на использовании алгоритмов вытесняющей многозадачности;
- переносимость системы, на уровне исходных кодов, за счет выделения ядра и написания ее основной части на языке высокого уровня;
- использование механизмов виртуальной памяти и свопинга для повышения уровня мультипрограммирования;

UNIX

- иерархическая файловая система, образующая единое дерево каталогов независимо от реального количества физических устройств, используемых для размещения файлов;
- использование простых текстовых файлов для настройки и управления системой;
- широкое применение командной строи;

UNIX

- унификация операций ввода-вывода на основе расширенного использования понятия «файл», логическое представление устройств и некоторых средств межпроцессного взаимодействия как «файлов»;
- организация сообщений и прерываний ввиде текста;
- использование конвейеров из нескольких программ, каждая из которых выполняет одну задачу, разнообразные средства взаимодействия процессов, в том числе и через сеть.

UNIX

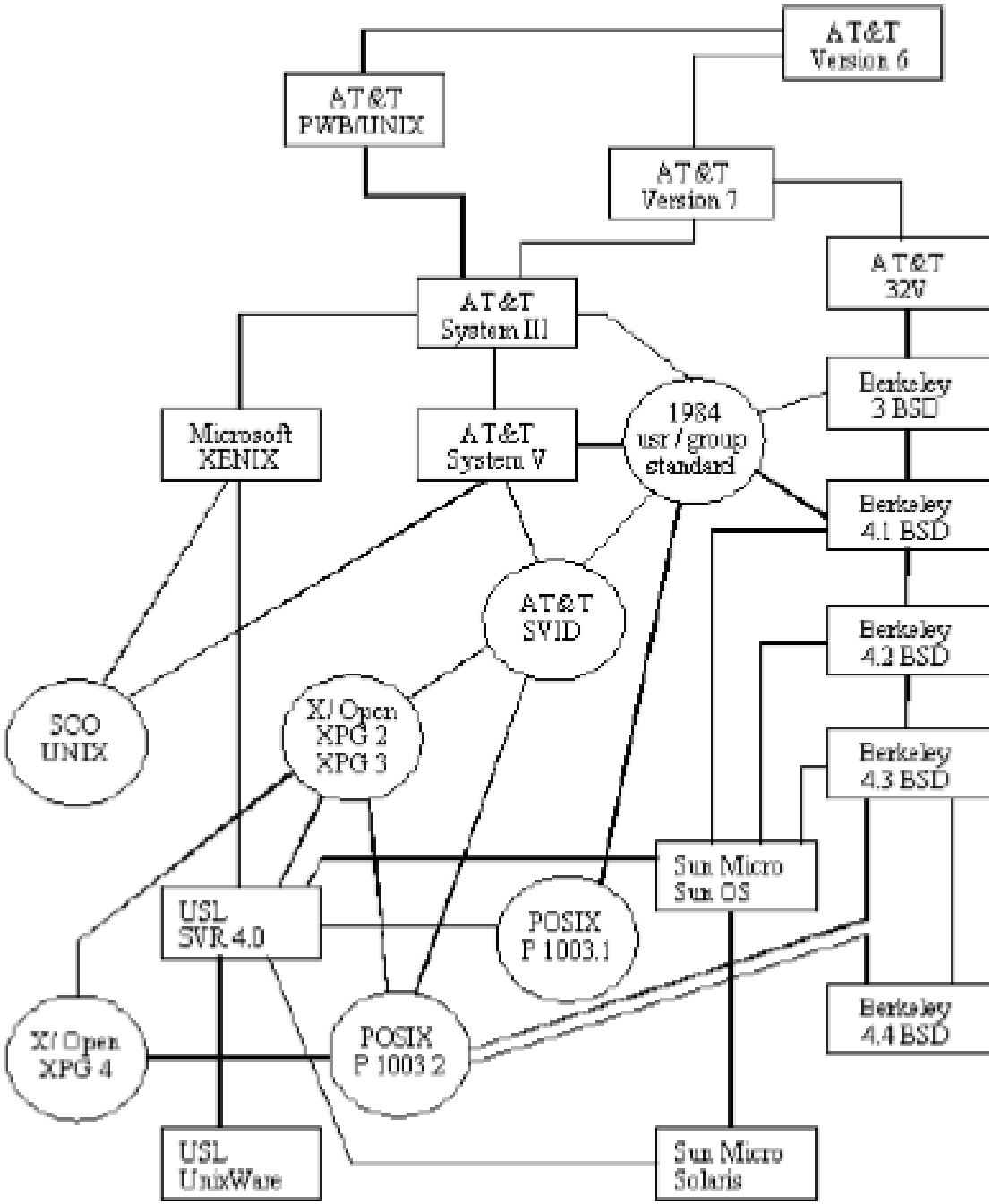


Рис. 14.3: Упрощенная схема взаимосвязий линий UNIX

Информационные технологии

- Скажите, пожалуйста, куда мне отсюда идти?

- Это во многом зависит от того, куда ты хочешь прийти, - ответил Кот .

- Да мне почти всё равно, - начала Алиса.

- Тогда всё равно, куда идти, - сказал Кот .

- Лишь бы попасть куда-нибудь, - пояснила Алиса.

-Не беспокойся, куда-нибудь ты обязательно попадёшь, - сказал Кот , - конечно, если не остановишься на полпути."

Льюис Керролл, “Алиса в стране чудес"

Технологии открытых систем

Когда мы пыт аемся выт ащит ь чт о-нибудь одно, оказывает ся, чт о оно связано со всем ост альным.

Закон Муира

- **Под основными свойствами открытых систем понимаются:**
- ***переносимость*** и ***переиспользуемость*** программного обеспечения, данных, моделей и опыта;
- ***масштабируемость*** как свойство сохранения работоспособности системы ИТ в условиях варьирования значений параметров, определяющих технические и ресурсные характеристики системы и/или поддерживающей среды.

26

Технологии открытых систем

- Под основными свойствами открытых систем понимаются:
- *интероперабельность*, т.е. возможность взаимодействия компонентов распределенной системы посредством обмена информацией и ее совместного использования;
- Открытость систем достигается на основе стандартизации их структуры и поведения, наблюдаемого на границах систем или их интерфейсах.

Парадигмы программирования

Приходится признать, что главная задача компьютерной науки — «не запутать все до неузнаваемости» — так и не была достигнута.

Увы, большинство наших систем слишком сложны, чтобы не тревожиться об их состоянии, они слишком хаотичны и запутаны, чтобы с ними можно было чувствовать себя уверенно и спокойно.

Обслуживание рядового заказчика в компьютерной отрасли находится на таком низком уровне, что пользователь постоянно ждет, что его система вот-вот рухнет. Мы являемся свидетелями массового, повсеместного распространения полного ошибок программного обеспечения, из-за чего нам должно быть очень стыдно.

Эдсгер Дейкстра

Мобильность программного обеспечения

Мобильность - это свойство программы, выражающееся в возможности ее адаптации для работы в различных окружениях.

Можно говорить о высокой или низкой мобильности программ, понимая при этом большую или меньшую степень легкости ее адаптации к вычислительной среде.

Высокая мобильность облегчает использование ранее созданного прикладного программного обеспечения и перенос программы с одной платформы на другую, что приводит к существенному уменьшению времени разработки и стоимости программного обеспечения.

Мобильность программного обеспечения

Мобильность в значительной степени зависит от распространенности используемого языка программирования, его стандартизации, точности и строгости эталонного описания, а также от имеющихся в языке специальных средств написания мобильных программ.

Мобильность во многом определяется также дисциплиной и технологией программирования, реализацией языка, качеством документации и др.

Достижение высокой мобильности программ может потребовать некоторого усложнения соответствующих алгоритмов. Степень усложнения тем меньше, чем лучше язык оснащен средствами поддержки мобильности.

Мобильность программного обеспечения

Стандартизация языков программирования создает предпосылки для повышения мобильности программного обеспечения. Использование международных стандартов языков программирования является одним из наиболее важных условий разработки мобильных программ для компьютеров любой архитектуры.

Вопросам стандартизации языков программирования во многих странах уделяется большое внимание, в этой деятельности участвуют многие ведущие фирмы, научные и учебные центры.

К сожалению, в нашей стране роль стандартов на языки программирования явно недооценивается. Высказывается иногда мнение, что стандартизация языка тормозит его развитие. Однако это утверждение ошибочно, так как в ISO (Международная организация по стандартизации) предусмотрен механизм периодического пересмотра стандартов.

Мобильность программного обеспечения

Что мешает:

- расширения стандарта, которые имеются не во всех реализованных версиях;
- различная степень контроля запретов и ограничений стандарта;
- различная трактовка неясных мест стандарта;
- различия в технических ограничениях (глубина вложенности, число параметров процедур и т.п.);
- различные способы представления данных;
- набор выделенных номеров для внешних устройств (экран, клавиатура, принтер и т.п.) и способ формирования имен файлов;
- использование процедур, которые отсутствуют в стандарте;
- разнообразие архитектур, ориентированных на параллельную обработку.

Рекомендации по мобильности программ

1. Главное требование при написании мобильных программ - строгое соблюдение стандарта. Это касается и программ, ориентированных на параллельную обработку.

В некоторых случаях стандарт не предписывает результаты выполнения программы, когда правила языка не выполняются, и конкретизация неопределенностей возлагается на разработчиков компиляторов. Программисты должны иметь представление о вытекающих отсюда проблемах и избегать ситуаций, которые четко не определены в языке. В большинстве случаев можно обойти эти трудности, запрограммировав соответствующее место другим способом. Однако программисты иногда прибегают к всевозможным "трюкам" и ухищрениям, используя особенности реализации конкретного компилятора, что, естественно, снижает мобильность программы. В то же время было бы неверным рекомендовать пользоваться в программах только средствами стандарта, особенно в тех случаях, когда программу не предполагается использовать в другой среде или когда используемые расширения имеются во всех тех окружениях, где будет выполняться данная программа.

Однако программист, создающий мобильную программу, должен знать стандарт и по возможности его придерживаться. В частности, если в конкретной реализации допускаются разные способы написания некоторой конструкции, рекомендуется использовать средства стандарта. Программист, использующий средства, которых нет в стандарте, должен понимать, что они могут отсутствовать при переносе его программы в другую вычислительную среду.

Рекомендации по мобильности программ

2. При написании мобильных программ следует выделить элементы зависимости алгоритма от компьютера и операционной системы (значения употребляемых констант, зависящих от среды, требуемую точность вычисления и диапазон данных, номера устройств ввода/вывода и др.).

3. В тех случаях, когда это необходимо, в операторах объявления типов следует использовать параметры типа.

4. Рекомендуется использовать имеющиеся в языке средства параметризации программ.

5. Если порядок вычисления выражений является существенным, программисту рекомендуется использовать скобки, указывающие последовательность выполнения операций.

6. Рекомендуется использовать средства условной компиляции (если в компиляторе они реализованы).

Рекомендации по мобильности программ

7. Программисты иногда ошибочно используют передачу управления внутрь цикла или в области цикла помещают операторы, изменяющие значения параметров цикла или управляющей переменной цикла. Такие ошибки не всегда диагностируются и могут привести к разным результатам в разных реализациях.

8. При написании мобильных программ программист не должен использовать устаревшие черты. Например, не рекомендуется использовать операторы GO TO со списком (GO TO по предписанию и GO TO по назначению).

9. Рекомендуется отмечать те места программы, где используются непараметризованные величины, зависящие от реализации, и/или расширения конкретной реализации. Это можно сделать с помощью специальных меток, либо с помощью комментариев или указать в документации. Это облегчит поиск таких мест, если понадобится адаптировать программу к новой вычислительной среде.

Два определения интероперабельности

«Классическое»

Интероперабельность

- способность двух или более систем обмениваться информацией и правильно использовать её

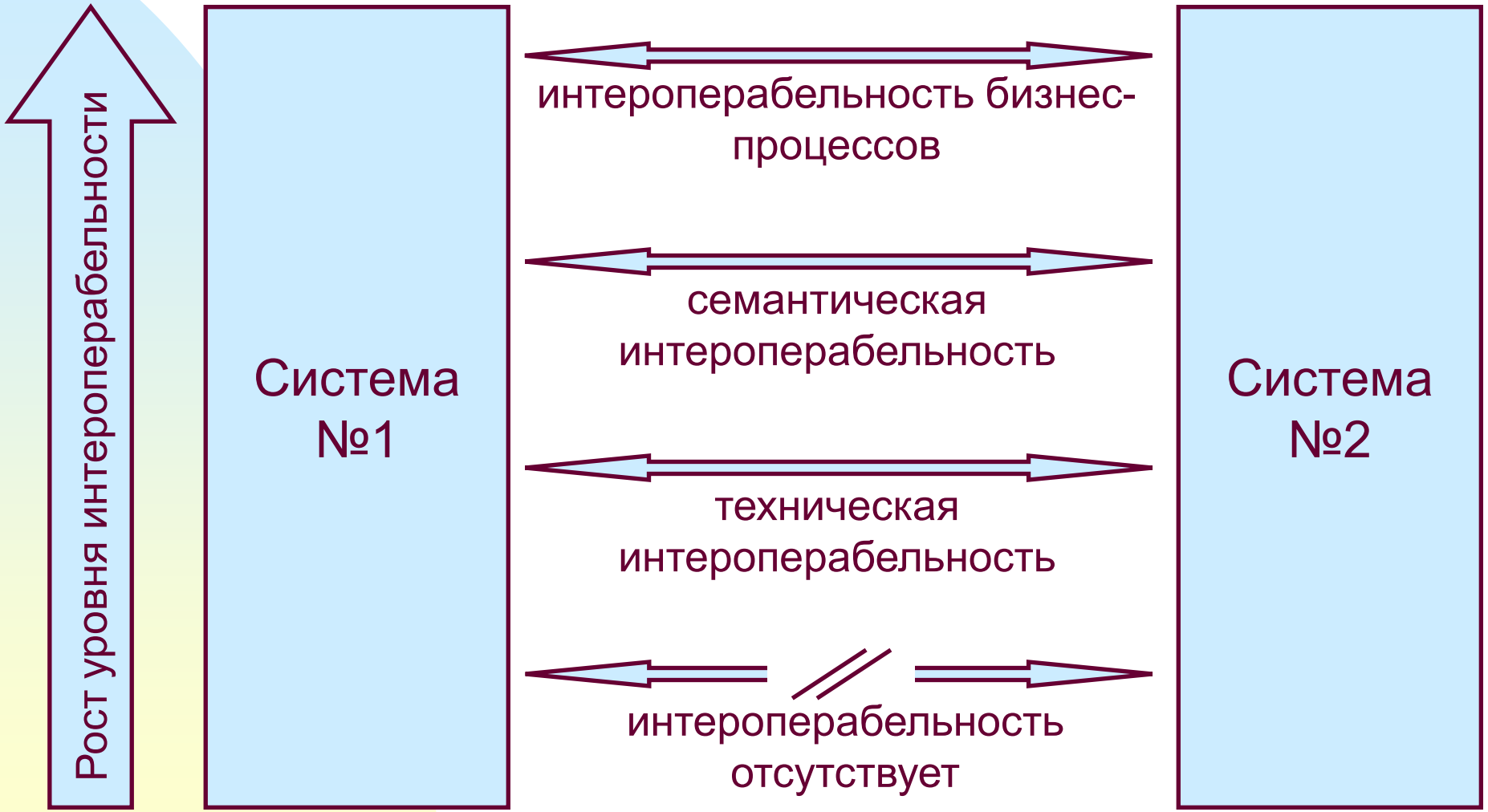
«Современное»

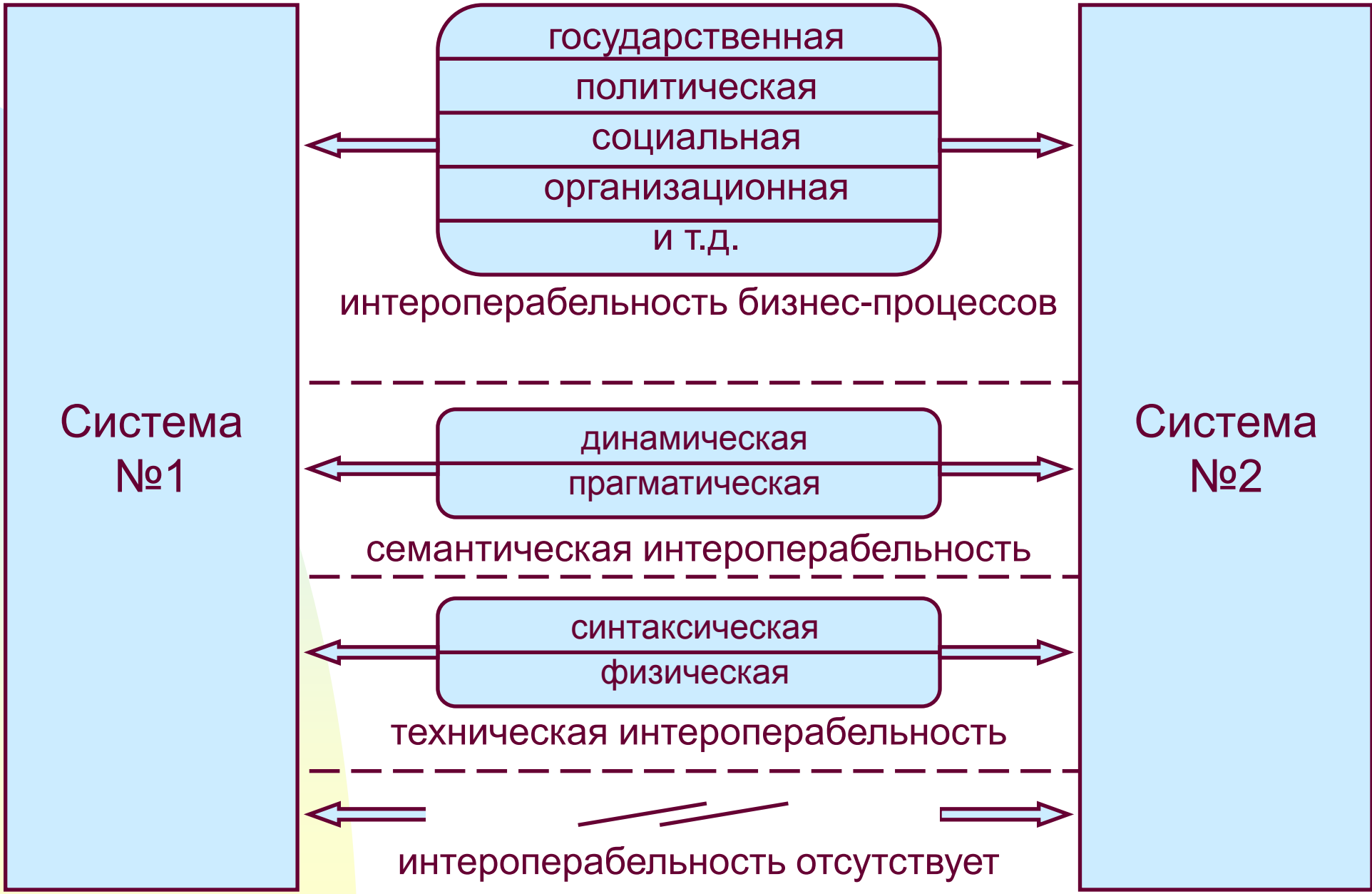
Интероперабельность

- способность различных систем и организаций к совместной работе

Кроме «технического» смысла появляется более широкий смысл, включающий социальные, политические и организационные факторы

Уровни итероперабельности





Модельная европейского обсуждения