Независимая
научно-практическая конференция
«Разработка ПО 2011»

Автоматизированная трансформация программ при миграции на новые библиотеки

В. Ицыксон, А. Зозуля

Санкт-Петербургский государственный политехнический университет

Лаборатория программно-аппаратных разработок

Миграция ПО на новые библиотеки

- Миграция на новую операционную систему
- Миграция на новую аппаратную платформу (например, мобильную)
- Переход на новые версии библиотек
- Переход на альтернативные библиотеки
- Перевод ПО на другой язык программирования
- …

Примеры задач миграции

- Миграция с Windows Threads на pthreads
- Переход с BSD socket на WinSock2
- Переход с использования библиотеки файлов на библиотеку потоков
- Замена опасных функций работы со строками на безопасные
- Портирование Fortran-программ на C
- Переход с GTK+ на QT
- Переход с CUDA на OpenCL
- Переход с DirextX на OpenGL

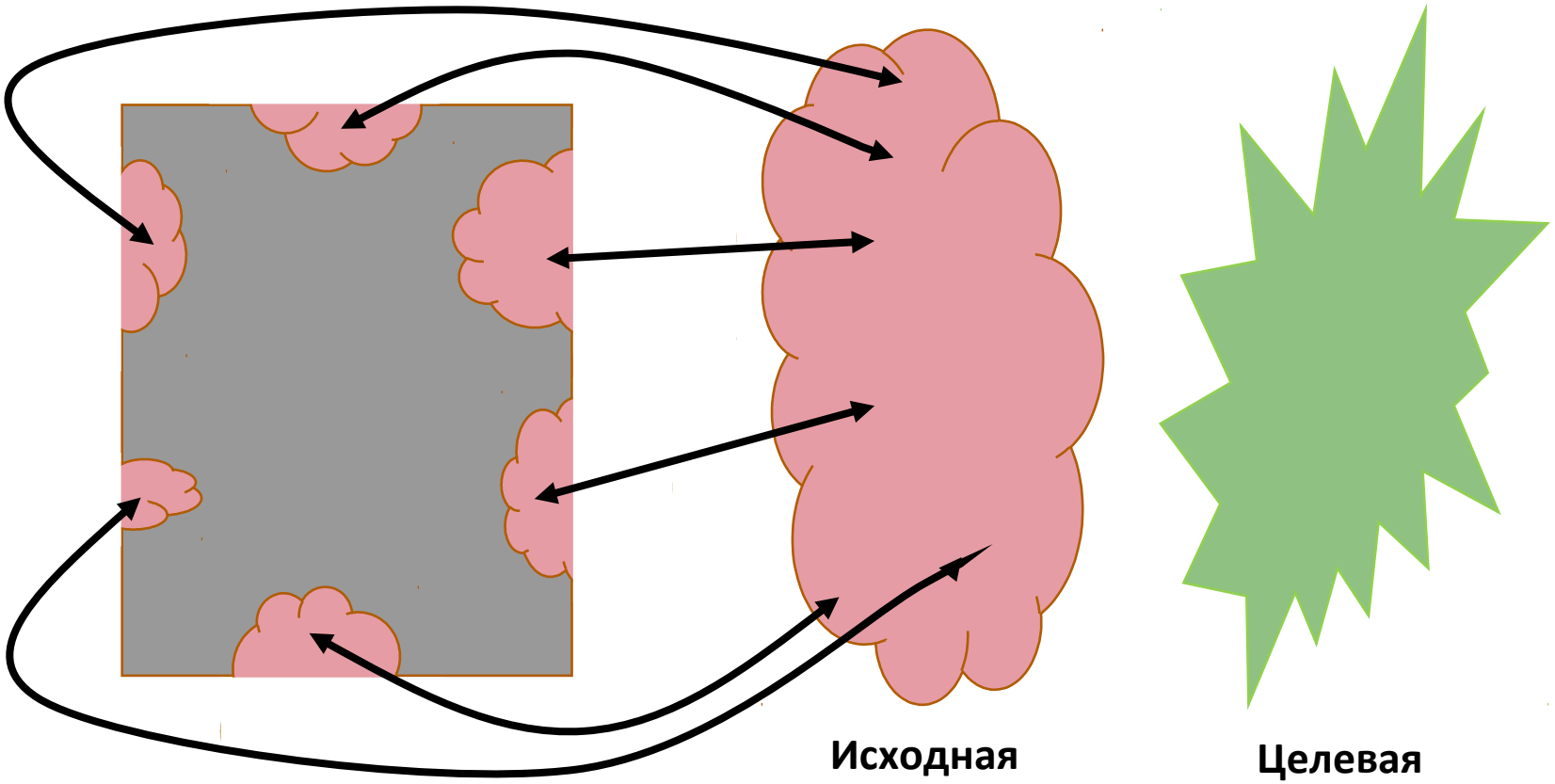
Как решается задача миграции?

- Подходы
  - Использование кроссплатформенных библиотек
  - Ручное переписывание вызовов
  - Использование параметризируемых макросов
  - Использование промежуточного слоя
- Проблемы
  - Множество ручных операций
  - Модифицированная программа – по сути – новая программа
  - Необходимость полного тестирования / верификации
- Решение
  - Автоматизация преобразования программ

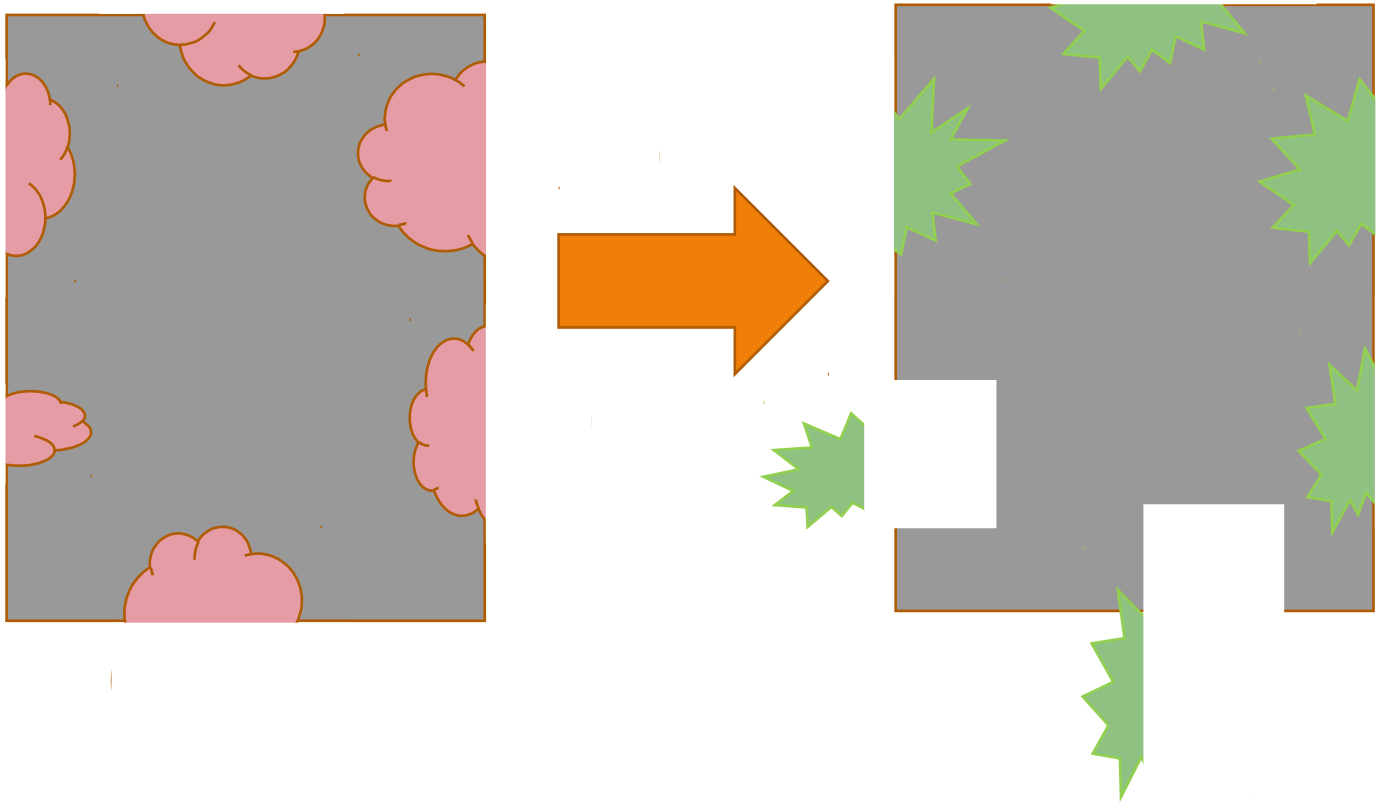
Связанные работы

- Проверка совместимости различных версий библиотек, ИСП РАН
- Анализ переносимости приложений между дистрибутивами, ИСП РАН
- Универсальные преобразователи текстов (TXL, Stratego/XT)
- Структурный реинжиниринг ПО (Rascal, DMS)
- Шаблонные трансформации программ
- Реинжиниринг программ, написанных на устаревших языках программирования, СПбГУ
- Межязыковые преобразования программ

Предлагаемый подход



Предлагаемый подход

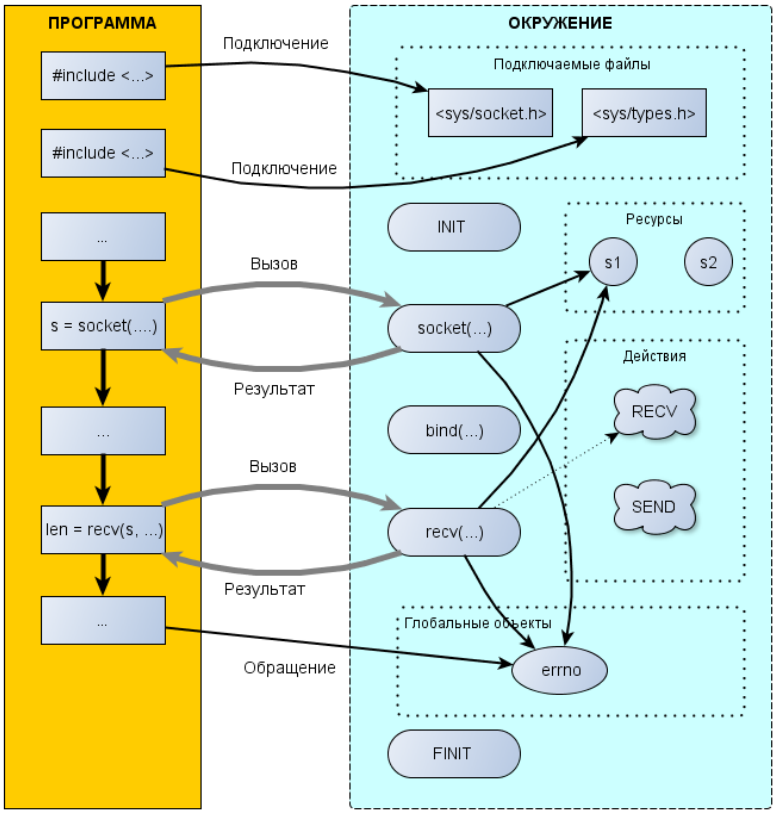


Основная идея

- Формализация структуры и поведения исходной и целевой библиотек с помощью спецификаций
- Анализ совместимости библиотек на основе совместимости спецификаций
- В случае совместимости:
  - формирование модели исходной программы
  - преобразование модели в соответствии со спецификациями библиотек
  - восстановление текста программы на основе модифицированной модели

Взаимодействие программы и библиотеки (язык С)

- Подключаемые файлы
- Вызовы библиотечных функций
  - Параметры функции
  - Возвращаемые значения
- Глобальные переменные
  - Объявленные в программе
  - Объявленные в библиотеке



Формализация описания библиотек

- Специфицируется не вся библиотека, а только ее **видимое** поведение – **частичная спецификация**
  - Описывает взаимодействие программы и библиотеки
  - Описывает побочные эффекты библиотеки
- Частичные спецификации задают семантику библиотеки
- Степень детализация описания определятся разработчиком
- Частичная спецификация задается на специализированном языке аннотаций PanLang

Элементы частичной спецификации библиотеки

- Подключаемые файлы
  - Спецификация заголовочных файлов библиотеки
- Глобальные объекты
  - Спецификация глобальных переменных, создаваемых библиотекой
- Ресурсы
- Семантические типы
- Семантические действия
- Описание поведения функций библиотеки
- Описание инициализации и финализации библиотеки

Ресурсы

- Модель объектов библиотеки или операционной системы, имеющих свой жизненный цикл
- Примеры ресурсов:
  - Файлы
  - Потоки
  - Сокеты
  - Нити
  - Семафоры
- Жизненный цикл ресурса описывается конечным автоматом
- Переход ресурса из одного состояния в другое осуществляется
  - при вызове функции библиотеки
  - по инициативе операционной системы

Семантические типы

- Расширяют типы языка программирования
- Используются для назначения абстрактной семантики объектам программы
  - Например, аргументам функции
- Используются для задания семантической интерпретации значений типа языка программирования

Семантические типы

Спецификация для Linux

```
semantic type
        SOCKET_TYPE(int);

semantic type
  SHUTDOWN_HOW(int) {
     READ:        SHUT_RD;
     WRITE:       SHUT_WR;
     READ_WRITE:  SHUT_RDWR;
};
function int shutdown(
 SOCKET_TYPE s, SHUTDOWN_HOW
how)
{
...
}
```

Спецификация для Windows

```
semantic type
        SOCKET_TYPE(SOCKET);

semantic type
  SHUTDOWN_HOW(int) {
     READ:        SD_RECEIVE;

     WRITE:       SD_SEND;
     READ_WRITE:  SD_BOTH;
};
function int shutdown(
  SOCKET_TYPE s, SHUTDOWN_HOW
how) {
...
}
```

Семантические действия

- Инкапсулируют побочное действие функции
- Задают абстрактную семантику функции
- Могут параметризироваться

```
action void SEND(SOCKET_TYPE socket);
action void RECV(SOCKET_TYPE socket);
```

- Используются в спецификациях поведения библиотечной функций

```
function SEND_RESULT send (SOCKET s, MSG, MSG_SIZE, FLAGS) {
   if (state(s) == $CONNECTED) {
      action SEND (s);
   }
}
```

Спецификация функций

- Описание сигнатуры функций
  - Спецификация параметров
  - Спецификация возвращаемого значения
- Описание поведения функции
  - Укрупненное описание алгоритма
  - Алгоритм описывается императивно
  - Описание состоит из:
    - Манипуляции с параметрами функции
    - Операций с глобальными переменные
    - Управляющих конструкций
    - Выполнения семантических действий
    - Манипуляций с ресурсами

Язык PanLang

- Используется для задания частичных спецификаций
- Совмещает декларативные и императивные описания
- Специальные ограничения на императивную часть
  - нет циклов
  - нет рекурсий
- Синтаксис похож на синтаксис языка C

Пример PanLang-спецификации библиотеки ввода-вывода (фрагмент)


```
requires
  <stdio.h>, <stdlib.h>;
semantic type
  FILE_TYPE (FILE *);
semantic type FLAGS (char*) {
  READ: "r";
  WRITE:"w";
  ALL: "rw";
}
semantic type HW_RES (int){
  ERR:[-inf; -1];
  OK:[0; +inf];
}
resource FILE_RES (FILE_TYPE){
  states OPEN, CLOSED;
  attribute FLAGS MODE;
}
action void READ(FILE);
```

```
function FILE fopen(
 FILE_NAME(char*), FLAGS mode){
  f = new FILE_RES(OPEN);
  attr(f, MODE) = mode;
  return f;
}
function HW_RES fread(
 BUFF, SIZE, PORTON p=1, FILE f)
{
  if (state(f) == $OPEN) {
    action READ(f);
    return $OK;
  }
  else {
    return $ERR;
  }
}
```

Создание частичной спецификации библиотеки

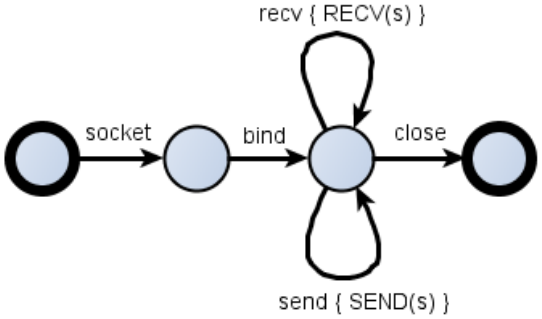
- Спецификации создаются разработчиком
  - идеально – разработчиком библиотеки
  - реально – разработчиком программы
- Исходная информация:
  - документация по библиотеке
  - понимание принципов работы библиотеки
- Шаблон спецификации может создаваться автоматически

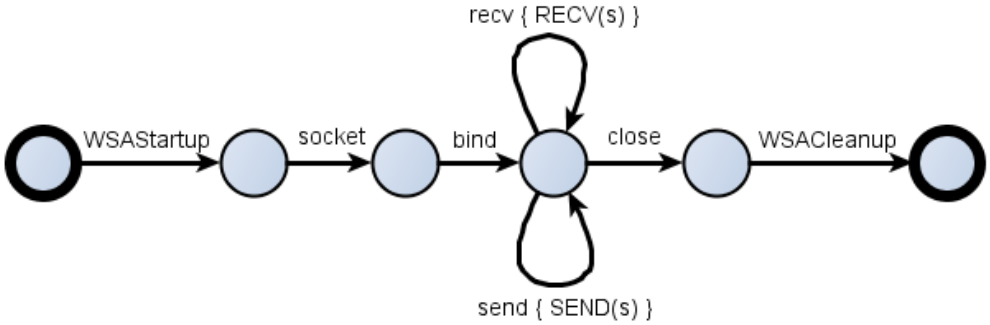
Проверка совместимости библиотек

- Две библиотеки совместимы, если реализуют одинаковое поведение
- Более точно:
  - Библиотека A может быть реализована средствами библиотеки B, если любая траектория поведения библиотеки A реализуется библиотекой B
  - Траектория библиотеки – последовательность событий, которая может быть сгенерирована библиотекой
  - События – семантические действия и (возможно) смена состояний ресурсов

Проверка совместимости библиотек

- Рассмотрим библиотеки BSD sockets и WinSock
- Поведение (упрощенное) библиотек описывается автоматами (в общем случае - системами автоматов):



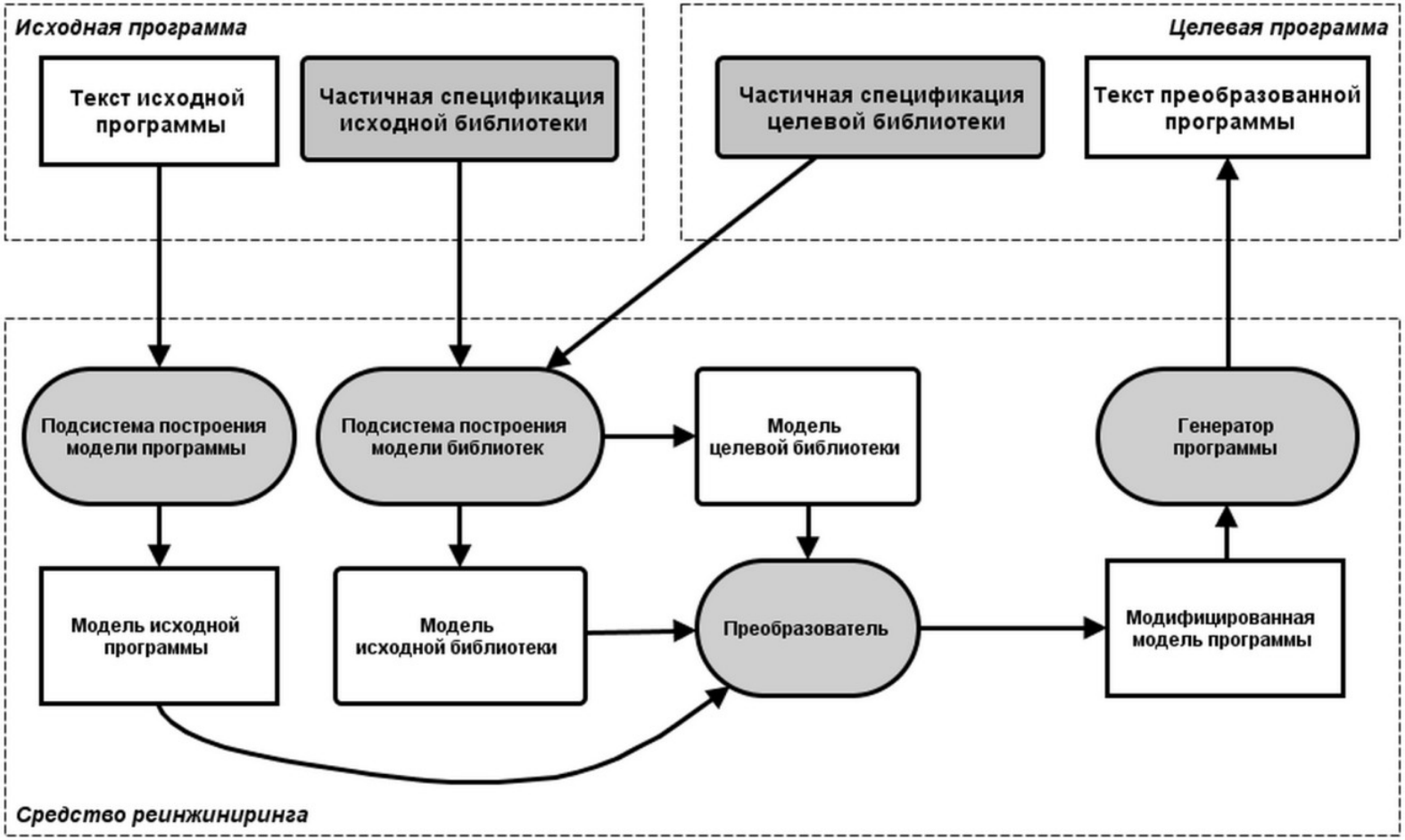


- Траектория библиотеки BSD sockets содержит конечную или бесконечную последовательно выполнения семантических действий RECV и SEND
- Очевидно, что поведение библиотеки WinSock содержит все такие траектории -> WinSock совместима с BSD sockets
- Более формально задача совместимости библиотек сводится к задаче проверки изоморфизма автоматов, описывающих поведение (побочные эффекты) двух библиотек

Трансформация программы

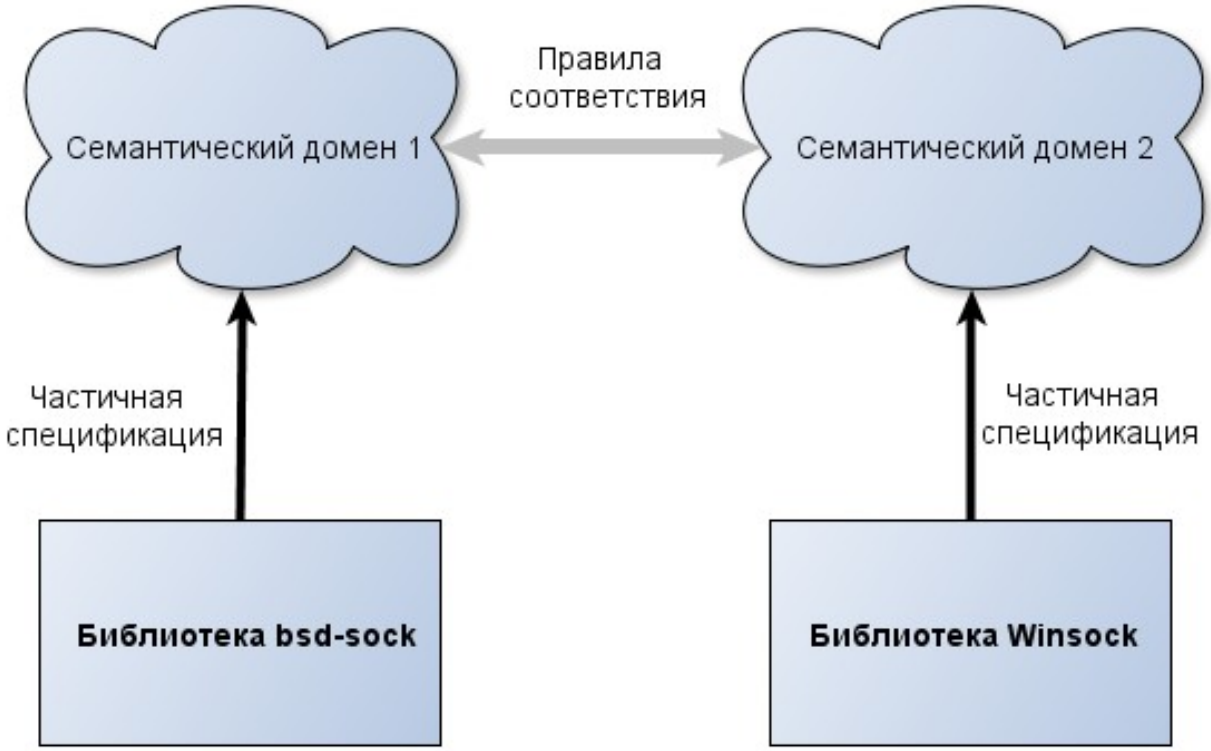
- В случае совместимости библиотек – реализуется алгоритм трансформации модели программы
  - разбор текста программы и формирование модели;
  - идентификация элементов модели, отвечающих за взаимодействие с библиотекой;
  - замена подключаемых файлов;
  - добавление в граф узлов, соответствующих функциям преобразования типов;
  - модификация узлов графа, соответствующих вызовам функций, в соответствии с сигнатурами функций, их семантикой и порождаемыми действиями;
  - модификация обращений к глобальным объектам
- Алгоритм трансформации использует информацию этапа проверки совместимости (соответствие трасс)
- По преобразованной модели строится код целевой программы

Прототип системы реинжиниринга



Об абстрактной семантике. Текущая реализация

Об абстрактной семантике. Многодоменная реализация



Ограничения прототипа системы реинжиниринга

- Работа с одним семантическим доменом
- Не поддерживаются
  - Сложные типы данных
  - Функции библиотеки, вызываемые по указателю
- Портирование однофайловых проектов

Планы по развитию подхода

- Поддержка многофайловых проектов
- Поддержка сложных типов данных
- Расширение языка для поддержки нескольких семантических доменов
- Расширение языка PanLang

Публикации

- **Ицыксон В.М., Зозуля А.В.** Формализм для описания частичных спецификаций компонентов программного окружения. Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. №4, 20 11. - СПб: Изд-во С. Петерб. ун-та. - 2011. - сс. 81-90.
- **V. Itsykson, A. Zozulya, M. Glukhikh.** Automated Program Reengineering When Porting Software to a New Environment Described by Partial Specifications . Proceedings of 2nd Workshop "Program Semantics, Specification and Verification". St.Petersburg-2011. pp. 111-119.
- **Ицыксон В.М., Глухих М.И.** Язык спецификаций поведения программных компонентов. Научно-технические ведомости СПбГПУ. Информатика. Телекоммуникации. Управление. №3, 2010. сс. 63-71. СПб: СПбГПУ

Контактная информация

- Санкт-Петербургский государственный политехнический университет
  - http://www.spbstu.ru
- Факультет технической кибернетики
  - http://ftk.spbstu.ru
- Кафедра компьютерных систем и программных технологий
  - http://kspt.ftk.spbstu.ru
- Лаборатория программно-аппаратных разработок
  - http://digiteklabs.ru

- Ицыксон Владимир Михайлович, доцент, к.т.н.
  - E-mail: vlad@ftk.spbstu.ru
  - Тел.: +7 (812) 297-22-38