Белорусский Государственный Университет
Факультет прикладной математики и информатики
Кафедра технологий программирования

Автоматизированная компоновка приложений служебно-ориентированной архитектуры

Подготовил:
Литвин Александр Викторович
Научный руководитель:
Войтешенко Иосиф Станиславович

2009 г.

Актуальность проблемы

С распростанением идеи индустрии программирования о введении «промышленной» **сборки** приложений из «**стандартных комплектующих**» наблюдается возрастание практического интереса к приложениям служебно-ориентированной архитектуры. Поэтому автоматизация бизнес-процессов – одна из важнейших научных и практических задач информатизации, во многом, возможно, определяющая будущее хозяйственное развитие.

Цель работы:

- Изучить эволюцию и систематизацию методологий программирования, **выделить** основные принципы и преимущества служебно-ориентированного программирования.
- **Изучить** языки описания высокого уровня.
- Ознакомиться с задачей компоновки служебно-ориентированных приложений. Изучить и **усовершенствовать** язык описания документооборота и автоматической компоновки приложений Entish.
- Проанализировать и **совместить** языки описания высокого уровня и компоновки приложений.
- **Реализовать** программную систему автоматической компоновки приложений.

Объект и методы исследования.

Объектом исследования являются различные протоколы обмена информацией, языки описания бизнес процессов высокого уровня, низкоуровневая реализация взаимодейсвия между потребителями и поставщиками служб, языки описания документооборота и автоматизированной компоновки приложений.

Основным **методом** решения поставленной проблемы является построение механизма отображения семантики служб и ее автоматизированного сопоставления с семантикой запросов клиентов путем интеграции широко распространенных технологий с ключевыми идеями, заложенными в некоторых зачаточных языках автоматизированной компоновки приложений служебно-ориентированной архитектуры.

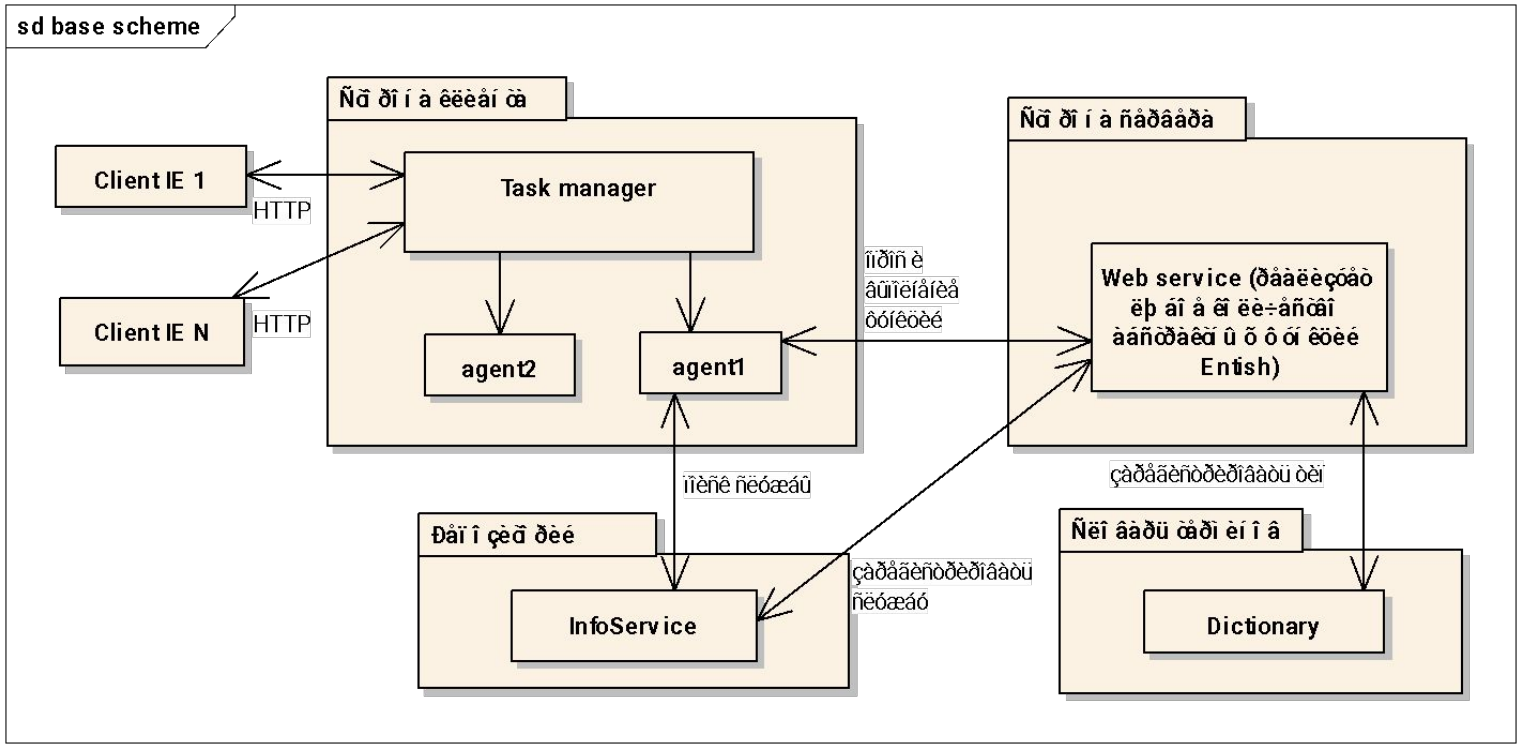
СОП и проблема компоновки

Автоматизация БП:

- Создание набора компонент (веб-служб)
- Организация взаимодействия

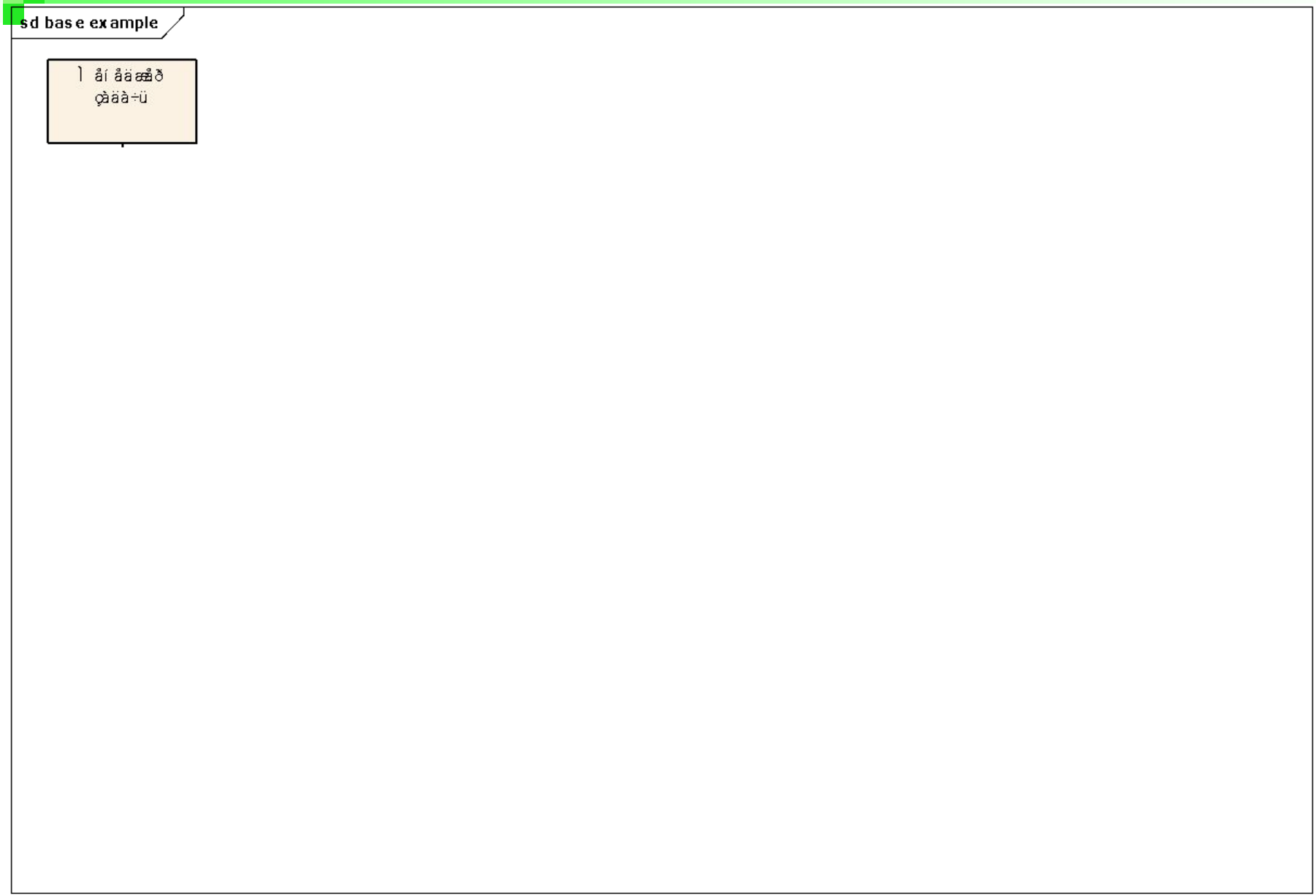
Компоновка веб-служб — нахождение набора атомарных служб, необходимых для реализации запроса пользователя, и определение порядка их выполнения.

Распределение компонент системы в соответствии с требованиями языка Entish



Основные составляющие языка Entish

- Фаза запроса:
  - Клиент указывает свою цель.
  - Сервис указывает, что для этого необходимо.
- Фаза выполнения:
  - Клиент создает входные данные.
  - Сервис скачивает данные и начинает работу.
  - Сервис высылает клиенту URI на результат.


sd base example
l åí åäæåð
çàää÷ü

Выявленные недостатки Entish

- Чрезмерное **резервирование** ресурсов.
- Огромные **объемы данных**, циркулирующие по сети.
- Нерешенная **проблема выбора** из множества поставщиков однотипных услуг.

Возможные улучшения Entish

- Разветвление плана работ (расширение схемы намерений агента)
- Обобщение на многошаговый бизнес-процесс
- Расширение на новую схему с инкапсуляцией трафика внутри ядра системы

BPEL и Entish

Для ликвидации существующих недостатков предлагается следующая организация взаимодействия BPEL и Entish: BPEL применяется "снаружи", то есть для конструирования сложного бизнес-процесса (и его выполнения), а Entish - "внутри", то есть для опрашивания текущего состояния услуги и получения ее согласия на участие в выполнении бизнес-процесса.

В BPEL необходимо разрешить привязку бизнес-процесса не к конкретным службам, а к абстрактным функциям. То есть заменить вызов жестко привязанных компонент, на поиск, опрос и привязку компонент, основанную на языке Entish (т.е. по абстрактной функции отыскивается подходящая служба, опрашивается на предмет готовности к взаимодействию, и только потом уже вызывается).

Заимствование идеи, а не протокола

Стандарт WSDL имеет один существенный недостаток: описание службы при помощи WSDL содержит только информацию об интерфейсе предоставляемых услуг и никоим образом не характеризует текущее **состояние** службы и ее **семантику**. Подобная проблема преодолевается в языке Entish путем проведения **опроса** вовлеченных в процесс взаимодействия респондентов. В то же время существенным недостатком языка Entish является чрезмерное **резервирование** ресурсов и использование **собственных** протоколов.

Привязка BPEL к абстрактной функции

- PartnerLinkType должен ссылаться на ядро системы
- Ядро системы должно реализовывать специальный HTTP обработчик

Цепочка связей между wsdl и bpel:

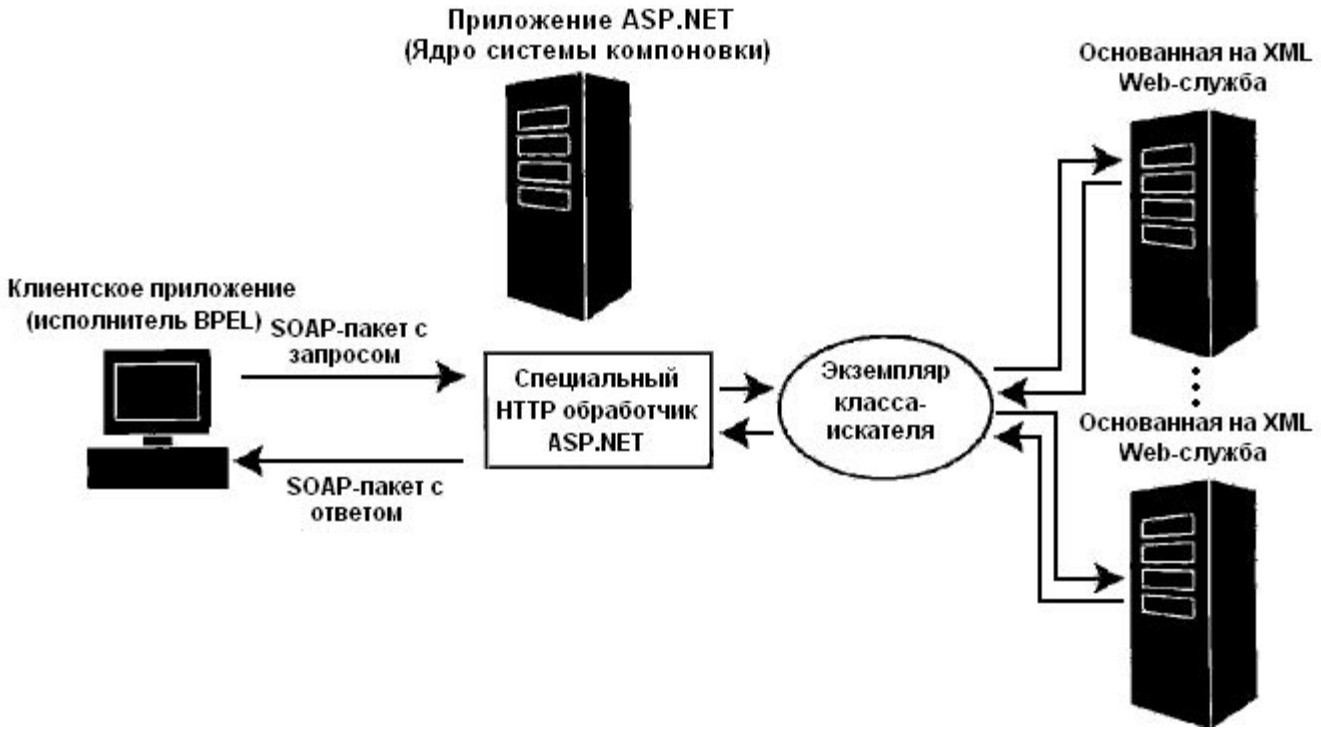
Service<->portType <->partnerLinkType <->partnerLink

Привязка BPEL к абстрактной функции

Пример ссылки на ядро системы:

```
<wsdl:service name="FileService">
  <wsdl:port name="FileServiceSoap"
    binding="tns:FileServiceSoap">
    <soap:address
    location="http://localhost:4085/Kernel/Rep.func" />
  </wsdl:port>
 </wsdl:service>
```

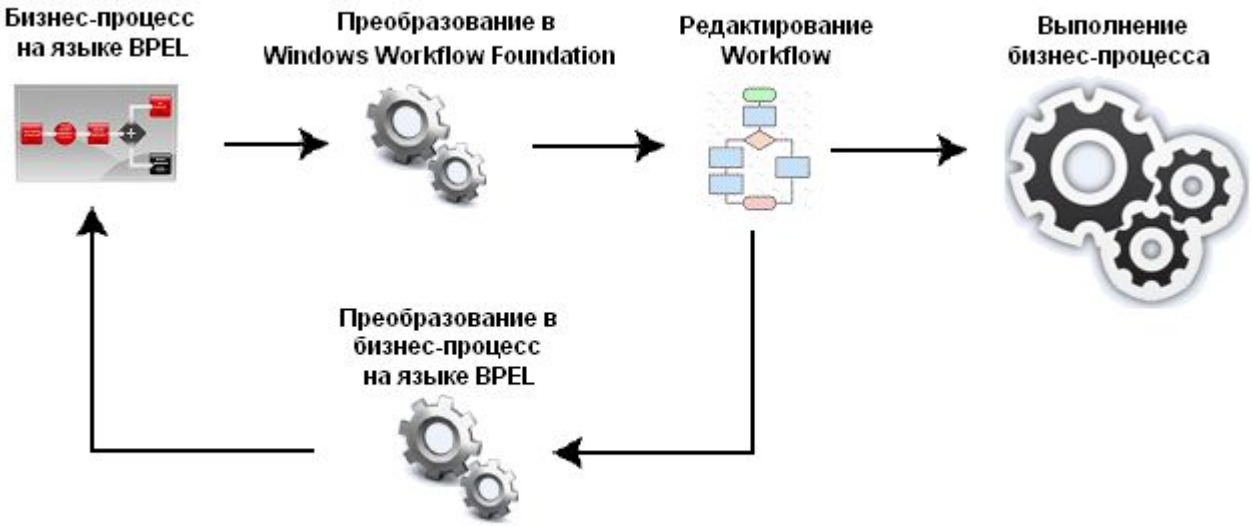
Концептуальная схема компоновки



Преимущества такой привязки

- Используются исключительно **стандартные** языки описания и протоколы
- Исполняющей средой может быть **любой стандартный сервер**, будь это BizTalk от Mircosoft или BPEL Process Manager от Oracle

Какого BPEL исполнителя использовать?



Что делает клиент и зачем он нужен?

- Предоставляет **доступный** исполнитель бизнес-процессов на языке bpel
- Облегчает **регистрацию** и **импортирование** описаний типов и функций
- **Расширяет** на Windows Workflow Foundation возможности автоматической компоновки

Диаграмма размещения

Основные результаты:

- **Исследована** эволюция и систематизация методологий программирования, **выделены** основные принципы и преимущества служебно-ориентированного программирования.
- Исследован и **усовершенствован** язык описания документооборота и автоматической компоновки приложений Entish.
- **Совмещены** языки описания высокого уровня и компоновки приложений.
- **Реализована** программная система автоматической компоновки приложений, которая доказывает жизнеспособность предложенных решений.

Спасибо за внимание!