САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

На правах рукописи

Ольхович Лев Борисович

РАСПАРАЛЛЕЛИВАНИЕ ДИРИЖИРУЮЩИХ БИЗНЕС-ПРОЦЕССОВ

05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

АВТОРЕФЕРАТ
диссертации на соискание учёной степени
кандидата физико-математических наук

Санкт-Петербург
2009

Работа выполнена на Кафедре системного программирования математико-механического факультета Санкт-Петербургского Государственного Университета.

Научный руководитель: доктор физико-математических наук, профессор Андрей Николаевич Терехов

Официальные оппоненты:

доктор технических наук, профессор
Тимофеев Адиль Васильевич
кандидат технических наук
Рощин Михаил Александрович

Ведущая организация:

Санкт-Петербургский государственный политехнический университет

Защита состоится “___”_____200_ г. в __ часов на заседании совета Д212.232.51 по защите докторских и кандидатских диссертаций при Санкт-Петербургском государственном университете по адресу: 198504, Санкт-Петербург, Петродворец, Университетский пр., 28, математико-механический факультет, ауд. 405.

С диссертацией можно ознакомиться в Научной библиотеке Санкт-Петербургского государственного университета по адресу: 199034, Санкт-Петербург, Университетская наб., д. 7/9.

Автореферат разослан " " _____ 200_ года

Учёный секретарь
диссертационного совета
доктор физико-математических наук, профессор И. К. Даугавет

Общая характеристика работы

Актуальность темы

В течение последних 10-15 лет быстрыми темпами развивается область автоматизации бизнес-процессов (БП) – в частности, благодаря всё возрастающей информатизации предприятий и всё большему проникновению информационных технологий в различные сферы человеческой деятельности. Если ещё недавно БП были не более чем «руководством к действию», правилами выполнения тех или иных процедур, то, с внедрением автоматизации, бизнес-процессы начинают непосредственно определять порядок функционирования предприятия.

Автоматизированные бизнес-процессы превращаются из абстрактных идей и «серебряной пули» менеджмента в действительность и становятся неотъемлемой частью автоматизированных систем управления предприятиями (АСУП) и информационных систем вообще. Бизнес-процессы обеспечивают беспрецедентную гибкость и позволяют адаптировать АСУП к изменяющемуся окружению и осуществлять взаимодействие с новыми видами систем без изменений программного кода и связанных с ним дополнительных затрат на обновление программного обеспечения (ПО). Соответственно, по мере всё большей автоматизации БП, увеличивается и влияние производительности БП на эффективность функционирования предприятий и организаций.

С учётом того, что время является критическим фактором в конкурентной борьбе, важной задачей становится обеспечение высокой производительности бизнес процессов. Распараллеливание бизнес-процесса, то есть выявление допускающих параллельное исполнение шагов в исходно последовательном бизнес-процессе, снижает время исполнения и тем самым повышает производительность бизнес-процесса.

Хотя распараллеливание программного обеспечения было и остаётся предметом активных исследований, исследований по распараллеливанию бизнес-процессов другими авторами не проводилось. При этом некоторые особенности основного типа применяемых сейчас бизнес-процессов — дирижирующих (реализующихся через вызовы внешних сервисов) бизнес-процессов в сервис-ориентированных окружениях — делают возможным полноценное распараллеливание, невозможное в случае «классического» ПО.

Цели работы

Целью работы является разработка метода повышения производительности дирижирующих бизнес-процессов при помощи их распараллеливания.

В рамках данной работы были созданы метод и реализующее его программное средство автоматизированного распараллеливания бизнес-процессов, рассмотрены вопросы интеграции распараллеливания в жизненный цикл БП и проведена апробация полученного технологического решения на промышленных бизнес-процессах.

Для проведения апробации был разработан метод генерации окружения БП для проведения нагрузочного тестирования БП при частичной недоступности используемых БП web-служб и создано программное средство, этот метод реализующее. По итогам апробации сделаны выводы об эффективности полученного технологического решения и, соответственно, разработанного метода автоматического распараллеливания БП.

Общая методика

Исследование проводилось в контексте методов распараллеливания и установления эквивалентности программного обеспечения, при этом оценивалась применимость существующих методов к бизнес-процессам. Для анализа влияния распараллеливания на производительность бизнес-процессов использовались общие методы оценки производительности бизнес-процессов; практические эксперименты проводились с применением методов тестирования производительности программного обеспечения. Для решения задачи распараллеливания применялись методы, основанные на методах анализа потоков данных программного обеспечения, используемые в компиляторах и анализаторах ПО.

Основные результаты

В настоящей диссертационной работе получены следующие основные результаты:

1. Проанализированы особенности предметной области — бизнес-процессы (БП) и их автоматизация с применением языка Business Process Execution Language (BPEL) и Web-служб и сделаны выводы относительно возможной эффективности автоматизированного распараллеливания таких БП.

2. На основании этих особенностей сформулирован критерий операционной эквивалентности дирижирующих БП.
3. На основании критерия эквивалентности и метода оценки производительности БП предложен структурный критерий оптимальности производительности (минимальности времени ответа) БП.
4. Предложен и исследован алгоритм оптимизации производительности БП при помощи их распараллеливания.
5. Создано инструментальное средство, реализующее предложенный алгоритм оптимизации.
6. Предложен новый вариант жизненного цикла БП, который включает в себя автоматизированную оптимизацию производительности БП.
7. Разработан метод генерации окружений БП исходя из их определений для нагрузочного тестирования БП в условиях частичной недоступности используемых web-служб.

Научная новизна

Все основные научные результаты диссертации являются новыми.

Практическая и теоретическая ценность

Практическая ценность настоящей диссертационной работы заключается в том, что:

1. предложенный в диссертации метод распараллеливания бизнес-процессов может быть использован:
   - для добавления функциональности распараллеливания в средства проектирования, редактирования и анализа бизнес-процессов;
   - для разработки средств анализа бизнес-процессов (в том числе интегрированных в системы мониторинга и исполнения бизнес-процессов) на основании описаний бизнес-процессов и историй их исполнения, автоматически анализирующих возможную эффективность их распараллеливания.
2. предложенный метод генерации окружения БП для проведения нагрузочного тестирования БП в условиях частичного отсутствия используемых web-служб может быть использован для добавления соответствующей функциональности в средства нагрузочного тестирования.

Теоретическая ценность настоящей работы заключается в предложенных критериях эквивалентности и оптимальности БП, алгоритмах распараллеливания БП, а также в том, что полученные результаты могут быть использованы в качестве отправной точки для дальнейших исследований в области оптимизации производительности бизнес-процессов, например, при изучении вопросов неэквивалентных преобразований бизнес-процессов (преобразований, изменяющих потоки данных) для повышения их производительности.

Апробация работы

Методы и полученные результаты данной диссертационной работы были оформлены в качестве заявки на выдачу патентных свидетельств (США, ЕС), а также реализованы как программное средство. Планируется применение основных результатов настоящей диссертационной работы в Siemens CT.

Результаты диссертации многократно докладывались на внутренних семинарах Siemens CT, вошли в результаты европейского проекта ASG, а также были были доложены на конференции International Conference on Internet and Web Applications and Services 2006, где доклад был отмечен дипломом «за лучшую статью».

Основные результаты диссертации изложены в 6 публикациях.

Публикации

Основные результаты работы изложены в 7 работах [1-7], перечисленных в конце автореферата, в том числе 2 работы [6,7] по перечню ВАК. Работы [1,2,5] написаны в соавторстве: в [1], Ольхович Л.Б. принимал участие в создании описываемого программного средства и проведении экспериментов; основной текст статьи принадлежит соавторам (A. Hennig, R. Wasgint, B. Petrovic). В [2], Ольхович Л.Б. является основным автором; Е. Рачинскому принадлежит авторство секции «Problem Statement» (стр. 2), а также участие в создании рис. 1 - 3; A. Hennig является автором части секции «Business Processes, Web Services and SOA» (стр. 2); Е. Рачинский и A. Hennig также осуществляли стилистические правки статьи. В [5], Ольховичу Л.Б. принадлежит раздел «5.1 iPPr» (стр. 29-36).

Структура и объем диссертации

Диссертация состоит из введения и 7 глав со сквозной нумерацией разделов, рисунков и таблиц. Текст диссертации изложен на 108 страницах. Список литературы содержит 90 наименований.

Содержание работы

Во **введении** показывается актуальность выбранной темы исследований, излагаются научный и исторический контексты диссертационной работы. Кратко перечислены основные результаты диссертации.

В **первой** главе охарактеризован контекст исследований и введены основные понятия предметной области, рассмотрены основные направления применения бизнес-процессов в информационных технологиях. Приведён обзор основных архитектур исполнения бизнес-процессов, описание *de-facto* стандартного языка спецификации дирижирующих бизнес-процессов — BPEL, а также удобный для проведения формального анализа бизнес-процессов формализм workflow-графов. В главе также дано краткое описание основных направлений исследований в области анализа производительности бизнес-процессов.

Согласно классическому определению, *бизнес-процесс* — это *набор действий и связей между ними, выполнение которых обеспечивает достижение какой-либо бизнес-цели*[1]. Бизнес-процессы в IT применяются для достижения двух целей: в случае *автоматизации внутренних бизнес-процессов* речь идёт об автоматизации информационной логистики и об интеграции гетерогенных информационных систем (Enterprise Application Integration, EAI) одного предприятия; в случае *автоматизации взаимодействия независимых партнёров* бизнес-процессы применяются как для автоматизации взаимодействия некоторой отдельной компании со своими контрагентами, так и для упорядочения и автоматизации совместной деятельности для всех её участников одновременно.

Можно выделить два основных типа *архитектур исполнения* бизнес-процессов: классическую и сервис-ориентированную. *Классическая* архитектура предназначена, в основном, для исполнения процессов, ограничивающихся отдельным предприятием; бизнес-процесс рассматривается как поток задач (*workflow*). В этом случае для обозначения среды исполнения бизнес-процесса используется термин "*workflow-система*". Типовая архитектура такой системы была предложена ещё в 1995 году консорциумом WfMC. *Сервис-ориентированная* архитектура (Service-Oriented Architecture, SOA)

1 На самом деле, бизнес-процессы делятся на "производственные", служащие непосредственно достижению бизнес-целей, и "вспомогательные", необходимые для выполнения производственных бизнес-процессов; более того, часто процент вспомогательных бизнес-процессов очень высок.

основана на оказании услуг (сервисов) независимыми поставщиками. Под услугой в данном случае понимается некоторый конкретный объём работ, выполняемый поставщиком услуг для достижения результата, желаемого потребителем. SOA основывается на идее слабой связи между программными агентами: услуги предоставляются потребителю независимыми поставщиками с использованием стандартных (универсальных) протоколов и интерфейсов. Сервисы формально независимы друг от друга; один и тот же сервис может предоставляться несколькими поставщиками. Использование SOA позволяет менять поставщика и реализацию сервиса без изменения самого процесса. Соответственно, появляются *дирижирующие* бизнес-процессы, реализующиеся через вызовы сервисов. Несмотря на то, что SOA не привязана к бизнес-процессам и её реализации существовали ещё задолго до появления самого понятия SOA, сейчас под SOA понимают, в основном, её реализацию с использованием Web-служб — набора протоколов, ориентированного на Internet.

Самым распространённым языком описания дирижирующих бизнес-процессов на настоящий момент является язык *BPEL*, который является результатом совместных усилий IBM и Microsoft (основан на языках XLANG и WSFL) и предназначен для описания процессов, в которых задействованы исключительно web-службы. Так как BPEL основан на идее дирижирования web-службами, то в нём не делается различия между сторонним процессом и просто web-службами: и те, и другие рассматриваются как *партнёры*, каждому из которых соответствует определённый интерфейс доступа. *Действия* (*шаги*) в BPEL могут быть простыми (неделимыми), так и составными, содержащими в себе другие действия (которые также могут быть составными). Соответственно, процесс представляет собой иерархию вложенных друг в друга действий.

BPEL является достаточно сложным языком, и, несмотря на его широкое промышленное применение, он не обладает формально заданной операционной семантикой. Поэтому для формальных построений мы будем использовать модифицированные *workflow-графы*, аналогичные *схемам программ*, предложенные Садик и Орловской задолго до появления BPEL. *Workflow-граф* — это направленный граф $P = (A, f, R, u)$, где:

- A — множество *вершин*, соответствующих *действиям;*
- $f: A \times A \to \{\top, \bot\}$ — *функция инцидентности;* дуге между вершинами a и b соответствует $f(a, b) = \top$;
- R — множество *ресурсов* процесса;

– $u: A \times R \to \{\mathbb{O}, \mathbb{R}, \mathbb{W}, \mathbb{C}\}$[2] — функция использования ресурсов.

Дуга $a \to b$ означает необходимость завершения действия a перед началом исполнения действия b. Соответственно, исполнению процесса соответствует последовательное исполнение всех действий на пути от старта к финишу; действия процесса могут во время своего исполнения использовать результаты уже исполненных действий, для этого они оперируют *ресурсами* R.

Большая часть работ по анализу производительности бизнес-процессов посвящена аналитическо-симуляционным исследованиям. В частности, для предсказания основного показателя производительности процесса — *времени ответа* — применяются методы СМО, вероятностная свёртка, симуляция (т.е. исполнение определения процесса в некотором виртуальном окружении) и редукция графа потока управления БП; также проводились исследования по нагрузочному тестированию бизнес-процессов. Мы будем использовать некоторые (основанные на правилах редукции графа потока управления БП) из этих методов для оценки влияния распараллеливания на производительность бизнес-процесса.

Во **второй** главе описываются существующие подходы к распараллеливанию «классического» программного обеспечения. Рассматривается представление программы с помощью формализма единичного статического присваивания и его применения к распараллеливанию ПО. Описываются вызванные невозможностью полноценного анализа ПО ограничения возможностей автоматического распараллеливания и даётся краткая характеристика проводимым в этой области исследованиям. Далее рассматриваются отличия дирижирующих бизнес-процессов от «классического» ПО.

Форма *единичного статического присваивания* (Single Static Assignment Form, SSA) — это представление программы, в котором каждое присвоение значения переменной приводит к созданию новой переменной. Такая форма чрезвычайно удобна для различных анализов в компиляторах, в том числе и при распараллеливании. Она позволяет легко обнаруживать зависимые операторы: так как значение каждой переменной присваивается только единожды, то читающий её оператор использует результат выполнения оператора, инициализировавшего переменную.

Любая прикладная программа содержит большое количество библиотечных и системных вызовов. Поэтому для её

2 Не используется, чтение, присвоение, изменение.

распараллеливания недостаточно проанализировать внутренние потоки данных: необходимо учитывать также состояния прикладных библиотек, операционной системы и, наконец, ЭВМ. Полноценный же анализ всех перечисленных факторов на этапе компиляции невозможен хотя бы ввиду различности конфигураций ПО и ЭВМ. Таким образом, теоретически невозможно и полноценное распараллеливание «классического» ПО, за исключением ПО с крайне ограниченной функциональностью. Соответственно, основные исследования связаны с *векторизацией циклов*, т.е. организации одновременного исполнения тела цикла для разных итераций, *локального распараллеливания*, т.е. определению независимости операторов, находящихся в непосредственной близости и не разделённых системно-библиотечными вызовами, и *спекулятивным распараллеливанием*, когда эквивалентность распараллеленной программы не гарантируется, а сама программа исполняется в некоторой специальной виртуальной машине, отменяющей результаты параллельного исполнения в случае возникновения конфликтов.

В отличие от «классического» ПО, в основе сервис-ориентированной архитектуры — SOA — лежит идея независимости потребителя услуги (сервиса) от её поставщика. Это означает, что любой используемый в бизнес-процессе сервис может быть заменён на аналогичный ему, но поставляемый другим поставщиком, а поставщик сервиса, в свою очередь, может в любой момент поменять реализацию сервиса. Это означает, в частности, отсутствие побочных эффектов у используемых процессами сервисов, что и позволяет производить полноценный анализ дирижирующих бизнес-процессов и, соответственно, их праспараллеливание.

В **третьей** главе рассматриваются влияние распараллеливания на производительность бизнес-процесса. Приводится формальная модель расчёта времени исполнения распараллеленного процесса. Рассматривается также практическая целесообразность распараллеливания с учётом повышенных накладных расходов на исполнение распараллеленного процесса. Приводятся и анализируются результаты практических экспериментов, подтверждающих целесообразность распараллеливания. Формулируются требования на распараллеливание бизнес-процессов, накладываемые особенностями их внедрения и использования. Наконец, рассматривается место распараллеливания в жизненном цикле БП.

Наихудшее (максимально возможное) время исполнения бизнес-процесса t_{max} равно сумме времён исполнения всех действий на

критическом пути $W = \{a_1,..., a_n\}$ в этом процессе: $t_{max} = t_W = \sum_i t(a_i)$. Влияние распараллеливания на наихудшее время исполнения БП зависит от местоположения распараллеливаемого участка:

1) распараллеливание шагов a_k и a_{k+1}, лежащих на критическом пути, в предположении $t(a_k) \geq t(a_{k+1})$, даст выигрыш $\Delta t_{max} = t(a_{k+1})$;
2) распараллеливание шагов, лежащих на субкритическом пути, не изменит t_{max};
3) распараллеливание шагов внутри тела цикла даст выигрыш $\Delta t_{max} \geq m * t'_C - t_C$, где m — максимально возможное количество итераций цикла, а t'_C и t_C — наихудшие времена исполнения распараллеленного и исходного тел цикла.

Аналогично определяется оценка *среднего* времени исполнения оптимизированного процесса в случае, когда известно распределение вероятностей $p(W_i)$ выбора того или другого пути исполнения W_i. Среднее время исполнения распараллеленного процесса $t'_{exp} = t_{exp} - \sum_i \Delta t(W_i) * p(W_i) < t_{exp}$

Следует заметить, что приведённые выше оценки не учитывают рост накладных расходов при параллельном исполнении шагов БП. Был проведён ряд экспериментов, в которых сравнивалось время исполнения последовательных и одновременных вызовов. Эксперименты подтвердили гипотезу о том, что выигрыш от распараллеливания перекрывает возрастание накладных расходов: причина заключается, в частности, в удалённых вызовах web-служб, выполнение каждого из которых требует, помимо вычислительной мощности системы, исполняющей БП, также и существенного времени на коммуникацию. Таким образом, можно сделать вывод, что и на практике распараллеливание не увеличивает времени исполнения процесса.

Распараллеливание БП – процедура, при проведении которой необходимо полностью исключить возможность внесения ошибки в бизнес-процесс. Соответственно, необходимо гарантировать корректность БП как при его создании, так и после внесения в него любого изменения, в том числе и связанного с распараллеливанием. Это требование накладывает определённые ограничения на допустимые методы распараллеливания: (1) все изменения должны производиться в терминах исходного языка высокоуровнего моделирования БП; (2) оптимизация должна производиться явно и отдельным шагом в процессе разработки/внедрения/поддержки БП. Для предотвращения негативных последствий от внедрения неоптимальных БП мы предлагаем производить распараллеливание БП ещё до его внедрения. В этом случае она будет производиться на фазе

реализации, сразу после создания исполняемого описания, а исполняться будет уже оптимизированный процесс.

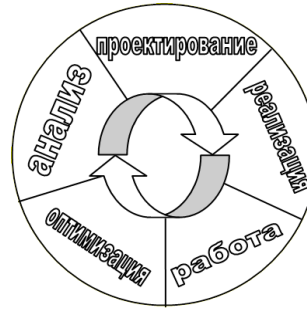

Рисунок 1. Жизненный цикл БП.

В **четвёртой** главе приводится формальная модель предлагаемого метода распараллеливания для workflow-графов. Определяется критерий операционной эквивалентности бизнес-процессов и достаточное условие эквивалентности с точки зрения упорядоченности исполнения шагов процесса. Далее формулируется структурный критерий эквивалентности и приводится способ построения распараллеленного процесса. Наконец, доказывается соответствие такого процесса критерию эквивалентности и его оптимальность при условии соблюдения данного критерия. В то время как существующие критерии оперируют понятиями состояния процесса либо окружения и не могут быть применены к БП с неопределённой семантикой операций либо игнорируют состояния данных процесса, представленные критерии предъявляют требования к потокам данных БП и, таким образом, применимы к БП с неизвестной семантикой операций.

Множество совокупных состояний процесса $S = S_a \times S_P \times S_e$, где S_a — множество состояний шагов процесса, S_P — множество состояний ресурсов процесса, и S_e — множество внешних состояний. Так как сервисы в дирижирующем процессе не могут иметь разделяемого состояния, то S_e можно исключить из рассмотрения. *Операционная семантика* σP процесса *P и* σa шага процесса *a* определяет то, как исполнение процесса или его шага влияет на совокупное состояние: $\sigma: S \to S$. Далее, если шаги в процесса исполняются в некотором порядке $\{ai\}$, то $\sigma P = \sigma a1 \circ ... \circ \sigma an$

Два детерминированных процесса P_1 и P_2 *операционно* эквивалентны, если $\sigma_{P1} = \sigma_{P2}$. Показано, что для эквивалентности двух

процессов, отличающиеся только допустимыми порядками выполнения действий $\{a_{i_k}\},\{a_{i_k}\}$, достаточно:

$$\forall a \in A \forall r \in (read(a) \cup change(a)) : init_{\{a_{i_k}\}}(a,r) = init_{\{a_{j_k}\}}(a,r)$$

где $init_{a_{i_k}} : A \times R \to A,\ init_{a_{i_k}}(a,r) = a_{i_{k_0}}$,

$init_{a_{i_k}}(a,r) = a_{i_{k_0}}, k_0 = \max\{k \mid k < \#(a), u(a_{i_k}, r) \in \{\mathbb{W}\mathbb{C}\}\}$

Далее предъявляется структурный критерий эквивалентности для процессов, заданных в виде workflow-графов, и показывается, что распараллеленный процесс $P^* = (A, f^*, R, u)$ можно построить по процессу $P = (A, f, R, u)$ следующим образом:

$$\forall a_1, a_2 \neq a_{start}, a_{finish} \in A: \quad f^*(a_1, a_2) = \quad \top \Leftrightarrow a_1 \in init^P(a_2)$$

$$\forall a \neq a_{start} \in A: \quad f^*(a_{start}, a) = \quad \top \Leftrightarrow \forall a' \neq a_{start}: f(a', a) = \bot \quad \text{(в } P^*\text{)}$$

$$\forall a \neq a_{finish} \in A: \quad f^*(a, a_{finish}) = \quad \top \Leftrightarrow \forall a' \neq a_{finish}: f(a, a') = \bot$$

В **пятой** главе описывается методика распараллеливания дирижирующих бизнес-процессов, основанная на результатах, описанных в главе 4. Приводится алгоритм распараллеливания, сохраняющий исходную иерархическую структуру контекстов бизнес-процесса.

Распараллеливание процесса, согласно результатам, описанным в предыдущей главе, производится следующим образом:

1. Для процесса строится его SSA-подобное представление.
2. Исходя из SSA-представления, строится граф зависимостей шагов процесса друг от друга.
3. Граф зависимостей сличается с исходным процессом и выявляются последовательно упорядоченные шаги, которые могут выполняться параллельно.
4. Процесс преобразуется таким образом, чтобы обнаруженные в пункте 3 шаги были упорядочены параллельно при сохранении эквивалентности процессов

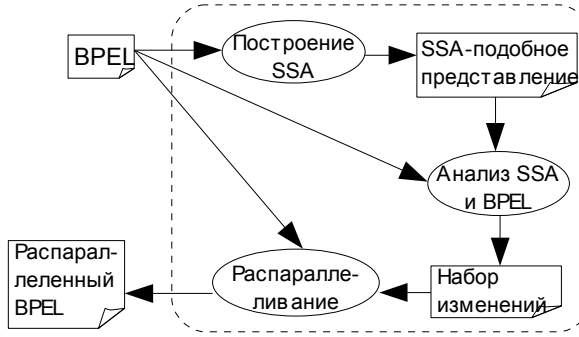

Рисунок 2. Распараллеливание БП.

Предложенный метод распараллеливания иллюстрируется на примерах типичного для крупных корпораций бизнес-процесса закупки оборудования и бизнес-процесса регистрации пассажира перед авиаперелётом.

В **шестой** главе анализируются требования к программной реализации, вытекающие из общих требований к распараллеливанию бизнес-процессов и его места в жизненном цикле БП. Далее приводится архитектура программного средства, реализующего распараллеливание, и кратко описываются основные аспекты реализации.

Исходя из места распараллеливания в жизненном цикле БП, а также из требований к самому процессу распараллеливания, можно определить следующие требования к этому программному средству: (1) оно должно быть совместимо со средством описания (средством создания исполняемых описаний) БП по формату входных и выходных описаний процессов; (2) оно должно интегрироваться в существующий жизненный цикл БП.

Всем этим требованиям отвечает средство распараллеливания БП, реализованное на языке Java 5.0 в качестве модуля расширения («plugin») для открытой платформы Eclipse. Оно основываться на EMF[3]-модели BPEL для загрузки, сохранения и обработки BPEL-процессов. Использование EMF-модели BPEL из редактора BPEL Editor for Eclipse позволяет интегрировать средство анализа с этим, а также с любым другим BPEL-редактором. EMF также используется для хранения в памяти SSA-представления. Соответственно, средство состоит из следующих компонент:

3 Eclipse Modeling Framework – инфраструктура моделирования и кодогенерации.

- компонента интеграции с пользовательским интерфейсом Eclipse;
- библиотека манипуляции SSA (на основе EMF);
- функциональная компонента, отвечающая за: построение сети анализа BPEL-процесса, анализ BPEL-процесса (с использованием сети анализа); построение набора требуемых изменений (на основании результатов анализа); выполнение собственно распараллеливания BPEL-процесса.

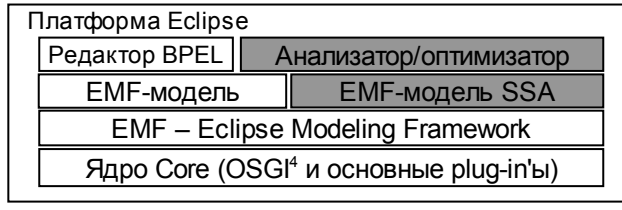

Рисунок 3. Компоненты средства распараллеливания.

В **седьмой** главе описан метод генерации окружения БП для проведения нагрузочного тестирования БП при частичной недоступности используемых процессом web-служб. Описано программное средство, этот метод реализующее.

В связи с необходимостью распараллеливания БП ещё на этапе их реализации, некоторые web-службы, используемые БП, могут быть недоступны. В такой ситуации невозможно проведение нагрузочного тестирования, необходимого для получения экспериментального подтверждения эффективности распараллеливания процесса и его соответствия предъявляемым требованиям по производительности.

Для того, чтобы сделать возможным нагрузочное тестирование на ранних стадиях создания БП, был разработан метод *генерации окружения* БП, заменяющий недоступные службы их сгенерированными заменителями. Метод базируется на подходе быстрого прототипирования производительности программного обеспечения «iPPr»[5], созданном в Siemens AG, CT SE 1 при непосредственном участии автора.

Генерация окружения БП производится iPPr автоматически на основе графических моделей окружения. Исходная модель окружения процесса автоматически создаётся по определению процесса и дальше уточняется пользователем для включения в неё, например, информации об используемых заменяемой web-службой ресурсах. В результате, на

4 Основанная на Java платформа, разработка Open Services Gateway Initiative.

5 Instant Performance Prototyping, мгновенное прототипирование производительности.

выходе пользователь получает окружение, полностью готовое к развёртыванию тестируемого БП.

В **заключении** сформулированы выводы и основные результаты работы.

Работы автора по теме диссертации

1. Hennig A. *Instant Performance Prototyping of EJB/J2EE Applications – A car rental example / R. Wasgint, B. Petrovic, L. Olkhovich* // Proceedings des 5. Workshop des Arbeitskreis PEAK, Munich, 2004. – Б. изд., 2004. – С. 29-36.
2. Olkhovich L. *Using Partly-Emulated Execution Environment for Measuring QoS-related Parameters of Business Processes / L. Olkhovich, E. Rachinsky, A. Hennig* // Proceedings des 6. Workshop des Arbeitskreis PEAK, Berlin, 2005. – Б. изд., 2005. – С. 9-18.
3. Olkhovich L. *Measuring QoS-related Parameters of Business Processes Using Partly-Emulated Execution Environment / L. Olkhovich* // Proceedings of Net.ObjectDays 2005 (ASG PhD Session), Erfurt, 2005. – Б. изд., 2005. – С. 97-110.
4. Olkhovich L. *Semi-Automatic Business Processes Performance Optimization Based On Redundant Control Flow Dependencies / L. Olkhovich* // Proceedings of the IEEE AICT/ICIW'06, Guadeloupe, French Caribbean, 2006. – Б. изд., 2006. – С. 146-153.
5. Kempter B. *Performance Engineering Methodology: ASG Public deliverable* [Электронный ресурс] / B. Kempter, L. Olkovich, E. Rachinskiy – Электрон. Дан. – Б. изд., 2007. – 52 с. – Режим доступа: http://asg-platform.org/, свободный.
6. Ольхович Л. *Автоматизированная оптимизация бизнес-процессов / Л. Ольхович* // Вестник Санкт-Петербургского Государственного Университета. Сер. 10: Прикладная математика, информатика, процессы управления. / СпбГУ. – С.: Изд-во СпбГУ, 2008. – №3 – С. 127-135.
7. Ольхович Л. *Оптимизация производительности бизнес-процессов при помощи их распараллеливания / Л. Ольхович* // Известия Волгоградского государственного технического университета: межвуз. сб. науч. ст. Сер. Актуальные проблемы управления, вычислительной техники и информатики в технических системах. / ВолгГТУ. – Волгоград: Изд-во ВолгГТУ, 2008. – № 5.– С. 73-75

ЛР № 040815 от 22.05.97

Подписано к печати --.--.2009. Формат бумаги 60х90 1/16.
Бумага офсетная. Печать ризографическая.
Объем 1 п.л. Тираж 100 экз. Заказ .
Отпечатано в отделе оперативной полиграфии НИИХ СПбГУ
с оригинал-макета заказчика.
198504, Санкт-Петербург, Старый Петергоф, Университетский пр., 2.