УДК 004.413

ИССЛЕДОВАНИЕ ПРОБЛЕМ ИНТЕГРАЦИИ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ УНАСЛЕДОВАННЫХ ПРИЛОЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ АСИНХРОННОГО ПИ-ИСЧИСЛЕНИЯ

А. Д. Брейман,
канд. техн. наук, доцент
А. Ю. Зерний,
аспирант
Б. В. Казьмин,
аспирант
Московский государственный университет приборостроения и информатики

Рассматриваются проблемы организации распределенных систем унаследованных приложений, использующих сторонние компоненты для управления интеграцией и последующим внутренним взаимодействием. Исследуется возможность устранения блокировок, которые возникают при взаимодействии приложений, применяющих разные типы связи. В качестве формальной модели исследования применяется асинхронное пи-исчисление. Предложен набор рекомендаций для организации рассматриваемых систем.

Ключевые слова *— интеграция приложений, промежуточное программное обеспечение, асинхронное пи-исчисление.*

Введение

Развитие технологий создания распределенных систем в последние годы во многом определено возрастающими требованиями бизнеса. Современные распределенные системы используют приложения сторонних поставщиков информационных услуг, которые реализуют типовые подзадачи более эффективно. Такой подход позволяет улучшить качество решения задач и минимизировать сопутствующие расходы. С развитием рассматриваемого подхода сформировалось новое направление в инженерии программного обеспечения (ПО), лежащего в основе таких систем.

Объектом нашего исследования является процесс интеграции гетерогенных приложений сторонних поставщиков информационных услуг в распределенную систему и управление их взаимодействием. При проектировании таких систем возникает необходимость создания универсальной архитектуры, позволяющей подключать приложения со скрытой внутренней организацией. Также возникает необходимость в инструментах адаптации рассматриваемого класса приложений для их использования в процессах проектируемой распределенной информационной системы.

Очевидно, что сторонние приложения можно использовать не только в качестве поставщиков специализированных услуг, но и для эффективного решения задач интеграции приложений и управления распределенной системой. Целью работы является исследование проблем блокировок при взаимодействии интегрируемых в распределенную систему унаследованных приложений, ориентированных на разные типы связи [1, 2], что в свою очередь поможет решить задачу автоматизации создания промежуточного ПО, направленного на управление взаимодействием в таких системах.

Требования к архитектуре

Сформулируем общие требования к архитектуре системы на основе интегрированных приложений, использующих для управления интеграцией и последующим внутренним взаимодействием сторонние сервисы:

1) управление интеграцией, реализация канала, трансляция данных, маршрутизация сообщений, хореография процессов и прочее управление взаимодействием должны быть вынесены в промежуточное ПО;

2) клиенты системы, т. е. приложения, интегрируемые в общую среду, должны быть слабо связаны с промежуточным ПО;

3) промежуточное ПО должно через соответствующие адаптеры предоставлять возможность взаимодействовать с клиентом на основе любого типа связи;

4) адаптер клиента должен минимально необходимо расширять функционал клиента;

5) взаимодействие «клиент — клиент» должно поддерживать все возможные модели связывания.

Интеграция на уровне приложения [2, 3] в рамках рассматриваемой архитектуры является исключительным случаем и требует подробного исследования. Отметим, что такая архитектура подразумевает связывание не приспособленных для того клиентов, комбинирование продиктованных их природой связей при необходимости соблюдать условия, накладываемые особенностями исследуемой области. На этом уровне интеграции возникают проблемы во взаимодействии друг с другом приложений, ориентированных на разные типы связи. Для того чтобы исследовать возможность устранения блокировок, которые, как будет показано далее, возникают между разносвязанными приложениями, необходимо рассмотреть используемые типы связи и построить математическую модель системы на основе таких типов.

Классификация типов связи

Исходя из особенностей исследуемой области можно утверждать, что интегрируемые компоненты не зависимы друг от друга и полностью реализуют свой функционал. При построении архитектуры в рамках рассматриваемого класса систем необходимо учесть следующее условие: промежуточное ПО с помощью соответствующих адаптеров должно обеспечивать взаимодействие клиентов, ориентированных на разные типы связи. Перечислить все существующие и перспективные технологии связи не представляется возможным, однако, как показано в работе [4], можно считать достаточным рассмотрение приведенных в работе шести типов связи:

1) сохранная асинхронная связь;

2) сохранная синхронная связь;

3) нерезидентная асинхронная связь;

4) нерезидентная синхронная связь с синхронизацией по приему;

5) нерезидентная синхронная связь с синхронизацией по доставке;

6) нерезидентная синхронная связь с синхронизацией по ответу.

Для исследования возможности взаимодействия интегрируемых приложений необходимо промоделировать парные сочетания приведенных типов связи.

Исследование сочетаний типов связи

Для исследования поведения взаимодействующих приложений предлагается использовать асинхронное пи-исчисление [5–7]. В статье [8] представлено формализованное описание процессов, отражающих приведенные в работе [4] типы связи, а также показано, что из 36 возможных сочетаний типов связи 12 сочетаний приводят к блокировке работы системы. В качестве примера возьмем пару типов связи, на основе которой, как будет показано далее, можно продемонстрировать вариант взаимодействия с блокировкой процессов.

Введем:

m_{AC} — сообщение, отправителем которого является процесс A, а получателем — процесс C;

m_{CA} — ссылка на процесс $C_{app.mess}$.

Отправитель (нерезидентная синхронная связь с синхронизацией по ответу)

$$A = (\nu\, m_{AC} u)\times$$
$$\times(\bar{x}u \mid uv.(\bar{v}m_{AC} \mid um_{CA}.m_{CA}s.m_{CA}s.A')),$$

где $A' = \tau_{A'}$ — процесс, следующий за синхронизацией взаимодействия.

Получатель (нерезидентная синхронная связь с синхронизацией по доставке)

$$C = (\nu cm)(C_{app} \mid C_{app.buff} \mid C_{app.mess}),$$

где $C_{app.buff}$ — процесс буфера; $C_{app.mess}$ — процесс счетчика состояния получателя; C_{app} — процесс получателя; C' — процесс, предшествующий обработке сообщения; C'' — процесс, следующий за обработкой сообщения:

$$C_{app.buff} = xu.(\nu v)\times$$
$$\times(\bar{w}w \mid vm_{AC}.(\nu d)(\bar{c}d \mid de.(\bar{e}m_{AC} \mid df.\bar{e}u)));$$

$$C_{app.mess} = (\nu qs)(!mp.(\bar{p}q \mid \bar{q}s));$$

$$C_{app} = C'.cd.(\nu e)(\bar{d}e \mid em_{AC}.(\nu p)\times$$
$$\times(\bar{m}p \mid pq.((\nu f)(\bar{d}f) \mid eu.\bar{u}q) \mid \tau_C.C''));$$
$$C' = \tau_{C'};\quad C'' = \tau_{C''}.$$

Для исследования возможностей совместной работы интегрируемых приложений с учетом сформулированных требований к архитектуре необходимо построить модель взаимодействия формализованных выше процессов.

Рассмотрим в качестве примера с блокировкой процессов пару «нерезидентная синхронная связь с синхронизацией по ответу — нерезидентная синхронная связь с синхронизацией по доставке». Для моделирования взаимодействия выбранных процессов с учетом описанных выше условий введем процесс, отражающий работу промежуточного программного обеспечения, выполняющего транслирующую функцию с учетом выбранных типов связи:

$$B = xu.(\nu v)(\bar{w}v \mid vm_{AC}.((\nu d)\times$$
$$\times \bar{y}d \mid de.(\bar{e}m_{AC} \mid dm_{CA}.\bar{u}m_{CA}))).$$

В результате процесс системы будет состоять из трех параллельных процессов:

$$S = (\nu xy)(A \mid B \mid C);$$

$$S \Rightarrow (\nu xyuvm_{AC}qs)(qs.A' \mid (\nu cdefmp)\times$$
$$\times (C'' \mid 0 \mid 0 \mid !mp.(\bar{p}q \mid \bar{q}s))) \equiv$$
$$\equiv C'' \mid (\nu qspm)(qs.A' \mid !mp.(\bar{p}q \mid \bar{q}s)) \sim$$
$$\sim (\nu qs)(qs.A') \mid C'' = S' \sim C'';$$
$$(x, y, u, v, m_{AC}, q, s) \notin fn(A');$$
$$(x, y, u, v, m_{AC}, q, s, c, d, e, f, m, p) \notin fn(C'').$$

Блокировка: процессу A для продолжения работы (перехода к процессу A') необходима синхронизация по $q \in bn(S')$; одновременно процесс C продолжает работу (процесс C''), при этом $q \notin fn(C'')$. Процесс S' сильно конгруэнтен процессу C'' (существует отношение сильной бисимуляции [9] между процессами S' и C''), вследствие чего процесс A' недостижим.

Устранение блокировок

Для выработки рекомендаций для построения рассматриваемого класса систем необходимо исследовать возможность решения выявленных блокировок.

Для обеспечения и управления взаимодействием приложений используется посредник — специальное ПО промежуточного уровня. Специфика рассматриваемого класса систем накладывает ограничения на изменения системы, а именно, исходя из сложности внесения изменений в компоненты, решение проблемы блокировок может быть достигнуто только благодаря изменению промежуточного ПО. Однако стоит отметить, что существуют исключительные условия принципиального характера, вынуждающие вносить минимальные изменения в интегрируемые компоненты.

Причиной возникновения блокировок при связывании компонентов рассматриваемого класса систем является невозможность осуществить синхронизацию отправителя (инициатора связывания) с получателем. В вариантах связываний с отправителями, работающими по правилам синхронной связи, отправитель сразу после отправки сообщения блокируется до получения извещения определенного типа от получателя. В случае, если получатель настроен на асинхронную или синхронную связь, но с извещением неподходящего типа, отправитель никогда не получит необходимое для продолжения работы извещение. Единственный вариант связывания с блокировкой и отправителем, работающим по правилам асинхронной связи, — связывание отправителя с асинхронной сохранной связью с получателем с нерезидентной асинхронной связью. Обеспечение сохранной связи с таким получателем невозможно. Причина та же самая: буфер отправителя, обеспечивающий сохранную связь, не может осуществить синхронизацию с получателем, а значит гарантировать отправителю доставку отправленного сообщения.

Определены три типа синхронизации — по приему, по доставке и по ответу. Типы синхронизации характеризуются моментом отправки извещения (сообщения, снимающего блокировку отправителя) получателем. Отправку извещения можно считать следствием смены состояния получателя, — перехода к обработке полученного сообщения или завершения обработки и перехода к обработке последующего сообщения. При этом извещение по ответу, отправляемое получателем в момент времени после доставки сообщения, является достаточным для снятия блокировки отправителя с синхронизацией по приему.

Для устранения выявленных блокировок необходимо обеспечить синхронизацию отправителя с получателем. С учетом ограничений рассматриваемого класса систем решение должно достигаться за счет изменения поведения посредника (ПО промежуточного уровня). Посредник должен гарантированно отправлять извещение, необходимое для синхронизации. При этом момент отправки извещения не должен опережать действительный переход получателя в соответствующее состояние. Таким образом, необходимым для устранения блокировки требованием к получателю является способность возвращать на запрос посредника состояние обработки сообщения.

Рассмотрим решение блокировки на примере связывания отправителя, работающего по правилам нерезидентной синхронной связи с синхронизацией по ответу, с получателем, поддерживающим нерезидентную синхронную связь с синхронизацией по доставке. Ранее было показано,

что такое связывание приводит к блокировке отправителя.

Отправителю, процессу A, для продолжения работы (перехода к процессу A') необходима синхронизация по $q \in bn(S')$; одновременно получатель, процесс C, продолжает работу (процесс C''), при этом $q \notin fn(C'')$. Процесс A', продолжение работы отправителя, недостижим.

Рассмотрим состояние системы, предшествующее синхронизации процесса C с процессом B:

$$S = (\nu xy)(A \mid B \mid C) \Rightarrow$$
$$\Rightarrow (\nu u)(um_{CA}.m_{CA}s.m_{CA}s.A' \mid (\nu d)\times$$
$$\times (dm_{CA}.\overline{u}m_{CA} \mid (\nu qmps)(\overline{d}q \mid \tau_C.C'' \mid \overline{q}s \mid !mp.(\overline{p}q \mid \overline{q}s)))) =$$
$$= S_1 \Rightarrow (\nu qs)(qs.A' \mid C'' \mid (\nu mp)(!mp.(\overline{p}q \mid \overline{q}s))) = S'.$$

Процесс S_1 — система S в состоянии, предшествующем синхронизации процесса C с процессом B. Процесс S' — система S в заключительном состоянии. Процесс A' недостижим.

Процессу A для продолжения работы (перехода к процессу A') необходима синхронизация по $q \in bn(S')$. Пусть $S'' = S_1$. Расширим S'' так, чтобы обеспечивалась необходимая синхронизация:

$$S'' = (\nu u)(A_1 \mid (\nu d)(B_1 \mid (\nu qmps)\times$$
$$\times (\overline{d}q \mid C_1 \mid \overline{q}s \mid !mp.(\overline{p}q \mid \overline{q}s))));$$
$$A_1 = um_{CA}.m_{CA}s.m_{CA}s.A';$$
$$B_1 = dm_{CA}.\overline{u}m_{CA}; \quad C_1 = \tau_C.C''.$$

Необходимую синхронизацию A_1 может обеспечить только процесс B_1. Расширим B_1 необходимым поведением:

$$B_1' = dm_{CA}.(\nu m_{CA})(\overline{u}m_{CA} \mid \overline{m_{CA}}s \mid \overline{m_{CA}}s).$$

Обеспечиваемая расширенным процессом B_1' синхронизация должна происходить после завершения обработки запроса процессом C_1. Добавим в процесс B_1' ожидание сообщения о достижении процессом C_1 необходимого состояния:

$$B_1'' = dm_{CA}.m_{CA}s.m_{CA}s.(\nu m_{CA})\times$$
$$\times (\overline{u}m_{CA} \mid \overline{m_{CA}}s \mid \overline{m_{CA}}s).$$

Необходимым требованием к получателю является способность к фиксации и возвращению состояния, достаточного для синхронизации; в рассматриваемом случае это состояние, соответствующее завершению обработки запроса. Интегрируемое приложение может реализовывать необходимое поведение, в противном случае требуется минимальное расширение унаследованного приложения. В рассматриваемом сочетании достаточно расширить поведение (или задействовать имеющийся функционал) получателя так, чтобы подпроцесс получателя C_1 возвращал состояние, соответствующее завершению обработки:

$$C_1' = \tau_C.(\overline{m}p \mid C'').$$

Подставим в S'' процесс B_1'' вместо B_1 и процесс C_1' вместо C_1:

$$S''' = (\nu u)(um_{CA}.m_{CA}s.m_{CA}s.A' \mid (\nu d)\times$$
$$\times (B_1'' \mid (\nu qmps)(\overline{d}q \mid C_1' \mid \overline{q}s \mid !mp.(\overline{p}q \mid \overline{q}s)))) \Rightarrow$$
$$\Rightarrow A' \mid C'';$$
$$(x, y, u, v, m_{AC}, q, s) \notin fn(A');$$
$$(x, y, u, v, m_{AC}, q, s, c, d, e, f, m, p) \notin fn(C'').$$

Процесс S''' (процесс S'' **с измененным поведением**) не имеет блокировок. Расширим процесс B в соответствии с B_1'':

$$B' = xu.(\nu v)(\overline{u}v \mid vm_{AC}.((\nu d)\times$$
$$\times \overline{y}d \mid de.(\overline{e}m_{AC} \mid dm_{CA}.m_{CA}s.m_{CA}s.(\nu m_{CA})\times$$
$$\times (\overline{u}m_{CA} \mid \overline{m_{CA}}s \mid \overline{m_{CA}}s)))).$$

Расширим процесс C_{app} в соответствии с C_1':

$$C_{app}' = C'.cd.(\nu e)(\overline{d}e \mid em_{AC}.(\nu p)\times$$
$$\times (\overline{m}p \mid pq.((\nu f)(\overline{d}f) \mid eu.\overline{u}q) \mid \tau_C.(\overline{m}p \mid C'')));$$

$$S_{final} = (\nu xy)(A \mid B' \mid (\nu cm)\times$$
$$\times (C_{app}' \mid C_{app.buff} \mid C_{app.mess})) \Rightarrow A' \mid C'';$$
$$(x, y, u, v, m_{AC}, q, s) \notin fn(A');$$
$$(x, y, u, v, m_{AC}, q, s, c, d, e, f, m, p) \notin fn(C'').$$

Система S_{final} не имеет блокировок. Проблема синхронизации решена расширением поведения посредника (B_1'') и минимальным расширением поведения получателя (C_1'). Аналогичным образом с помощью предложенного подхода решаются выявленные блокировки возможных сочетаний типов связи [8].

Заключение

В настоящей работе были исследованы проблемы интеграции компонентов в распределенных системах, использующих для управления интеграцией и последующим внутренним взаимодействием сторонние сервисы. Были сформу-

лированы требования к архитектуре такого класса систем, а также описаны возможные типы связи интегрируемых в систему компонентов. Для исследования возможностей совместной работы интегрируемых компонентов с помощью асинхронного пи-исчисления была построена математическая модель взаимодействия компонентов, поддерживающих описанные типы связи. В результате моделирования выявлены варианты с блокировкой процессов [8]. В завершение работы на математической модели системы было показано решение выявленных блокировок. На основе проведенного исследования можно сделать вывод, что взаимодействие компонентов, ориентированных на разные типы связи, в системах рассматриваемого класса, удовлетворяющих поставленным требованиям, при условии предложенных изменений возможно, а следовательно, возможно решение задачи автоматизации создания промежуточного программного обеспечения, направленного на управление взаимодействием в таких системах.

Литература

1. **Гофф М. К.** Сетевые распределенные вычисления. Достижения и проблемы. М.: Кудиц-Образ, 2005. — 320 с.
2. **Ладыженский Г.** Интеграция приложений такая, как она есть // Открытые Системы. 2006. № 9. http://www.osp.ru/os/2006/09/3776484/.
3. **Хоп Г., Вульф Б.** Шаблоны интеграции корпоративных приложений: Проектирование, создание и развертывание решений, основанных на обмене сообщениями. — М.: Вильямс, 2007. — 672 с.
4. **Таненбаум Э. С., Ван Стеен М.** Распределенные системы. Принципы и парадигмы. — СПб.: Питер, 2003. — 877 с.
5. **Milner R.** Communicating and mobile systems: the pi-calculus. — Cambridge: Cambridge University Press, 1999. — 161 p.
6. **Boudol G.** Asynchrony and the pi-calculus: Research Report 1702 / INRIA. — Sophia-Antipolis, 1992. — 11 p.
7. **Honda K., Tokoro M.** An object calculus for asynchronous communication // Proc. of ECOOP'91. Geneve, 1991. P. 133–147.
8. **Брейман А. Д., Зерний А. Ю., Казьмин Б. В.** Использование асинхронного пи-исчисления для исследования моделей связи при решении задач организации распределенных систем // Вестник МГУПИ. М.: МГУПИ, 2009. С. 57– 63.
9. **Amadio R. M., Castellani I., Sangiorgi D.** On Bisimulations for the Asynchronous pi-calculus // Theoretical Computer Science. 1998. Vol. 195. N 2. P. 291–324.