НАУЧНЫЙ ОБЗОР

УДК 004.4’2
DOI 10.17513/snt.40178

ТЕХНОЛОГИЯ ИНТЕГРАЦИИ И ТЕСТИРОВАНИЯ АВТОМАТИЗИРОВАННЫХ СИСТЕМ

Коданев В.Л., Федин Ф.О., Грачёва Е.В.

ФГБОУ ВО «МИРЭА – Российский технологический университет», Москва,
e-mail: kod_v@mail.ru, nidef@mail.ru, katetcpip@yandex.ru

Цель исследования – разработка технологии, направленной на упрощение процессов интеграции и отладки компонентов автоматизированной системы, повышения их надежности и производительности. На основе системного анализа и использования промежуточного программного обеспечения (программ-заместителей) в статье исследуется задача разработки технологии, направленная на автоматизацию и упрощение процессов обнаружения ошибок в программных компонентах, взаимодействующих по сетям. Предлагаемое решение ориентировано на использование программ-заместителей для облегчения взаимодействия между различными компонентами системы, тем самым повышая эффективность взаимодействия функционального программного комплекса внутри автоматизированных систем посредством таких программ. Формулируется гипотеза о существенном преимуществе при мониторинге, регистрации и изменении сетевого трафика при применении программ-заместителей. Такой подход не только облегчает выявление неточностей и ошибок в работе компонентов, но и позволяет проверить их работоспособность в различных условиях. Внедрение этой технологии обеспечивает повышенную гибкость и адаптируемость системы на этапах ее разработки и эксплуатации, способствует сокращению времени отладки и интеграции и в конечном итоге повышает общую надежность и производительность информационных и управляющих систем. Эта технология направлена на оптимизацию процессов интеграции и отладки компонентов системы, повышение их надежности и производительности за счет упрощения процессов поиска ошибок во взаимодействующих с сетью программных компонентах. Результаты исследования предоставляют разработчикам программного обеспечения автоматизированных систем широкий спектр возможностей для детального мониторинга взаимодействия, регистрации данных и настройки трафика, что позволит осуществлять комплексный анализ и тестирование системы.

Ключевые слова: информационная система, интеграция, тестирование, отладка, технология, заместитель, посредник, сетевое взаимодействие

INTEGRATION AND TESTING TECHNOLOGY AUTOMATED SYSTEMS

Kodanev V.L., Fedin F.O., Gracheva E.V.

MIREA – Russian Technological University, Moscow,
e-mail: kod_v@mail.ru, nidef@mail.ru, katetcpip@yandex.ru

The goal of the work is to develop technology aimed at simplifying the processes of integration and debugging of automated system components, increasing their reliability and performance. Based on system analysis and the use of middleware, the article explores the task of developing technology aimed at automating and simplifying processes detecting errors in software components interacting over networks. The proposed solution is focused on the use of proxy programs to facilitate interaction between various system components, thereby increasing the efficiency of interaction of a functional software package within automated systems through such programs. A hypothesis is formulated about a significant advantage in monitoring, recording and changing network traffic when using proxy programs. This approach not only makes it easier to identify inaccuracies and errors in the operation of components, but also allows you to check their performance under various conditions. The implementation of this technology provides increased flexibility and adaptability of the system at the stages of its development and operation, helps reduce debugging and integration time, and ultimately increases the overall reliability and performance of information and control systems. This technology is aimed at optimizing the processes of integration and debugging of system components, increasing their reliability and performance by simplifying the process of finding errors in software components interacting with the network. The results of the study provide software developers of automated systems with a wide range of opportunities for detailed monitoring of interactions, data recording and traffic configuration, which will allow for comprehensive analysis and testing of the system.

Keywords: information system, integration, testing, debugging, technology, substitute, intermediary, network interaction

Введение

Применение специального прикладного программного обеспечения позволяет автоматизировать рабочие процессы и тем самым повысить эффективность функционирования предприятия. В то же время интеграция специальных прикладных приложений в современных информационных и управляющих системах неизменно выявляет многочисленные ошибки, несоответствия и проблемы из-за их сложности и множества задействованных компонентов.

Существует несколько способов взаимодействия приложений: каждый с каж-

дым; на уровне пользовательских интерфейсов; на уровне данных; на уровне информационных ресурсов; на уровне корпоративных приложений; веб-сервисы и с помощью промежуточного программного обеспечения – программ-заместителей.

Многие организации не могут отказаться от уже налаженной рабочей сети. Интеграция новых приложений может нарушить рабочий процесс. В этом случае формируется специальный интерфейс – программа-заместитель, который становится своеобразным мостом между рабочей системой предприятия и новыми необходимыми приложениями. Рассматриваемая в статье технология позволит автоматизировать и упростить процессы поиска ошибок в программных компонентах, взаимодействующих по сети.

Если требуется интегрировать одно или два приложения в автоматизированную систему, то можно использовать метод «точка-точка». При таком методе программы просто объединяются. Ключевой момент при таком методе интеграции – четкое понимание того, как именно системы будут обмениваться данными. Для интегрируемых приложений просто создается общий модуль, через который и будут взаимодействовать программы. Интегрируются программы путем их записи в общую базу данных приложений или через API.

При использовании метода «сервисная шина» появляются практически неограниченные возможности масштабирования системы, гибкость, централизация контроля и возможность интеграции с другими системами.

Цель исследования – на основе системного анализа и использования промежуточного программного обеспечения (программ-заместителей) разработать технологию, направленную на упрощение процессов интеграции и отладки компонентов системы, повышения их надежности и производительности.

Материалы и методы исследования

Разрабатываемые в настоящее время информационно-управляющие системы отличаются высокой сложностью, большим количеством компонентов, создаваемых или функционирующих в гетерогенной среде. Часто трудно даже наглядно отобразить основные связи между элементами системы – комплексами функциональных программ (КФП). Учитывая большое количество каналов информационного взаимодействия между КФП, для организации взаимодействия в этом случае необходимо использовать специальные средства, повышающие эффективность планирования, конфигурирования и мониторинга сетевых взаимодействий.

Дополнительные сложности возникают при разработке, интеграции, отладке и тестировании таких КФП. Многочисленные ошибки и недочеты, которые необходимо устранить, вызываются разными факторами: различные команды исполнителей с отличающимися методами, приемами работы и компетенциями; недостатки описания протоколов взаимодействия; неполнота и/или несовершенство методик автономных испытаний КФП и др.

Решение проблемы организации надежного взаимодействия стало основанием для разработки большого количества шаблонов программной архитектуры и технологий, к числу которых, например, можно отнести:

– низкоуровневые технологии (сокеты, RPC) [1];

– сервисно-ориентированные архитектуры (SOA, WebServices) [2];

– технологии взаимодействия распределенных объектов (RFPS, CORBA, COM, DCOM) [1, 2];

– клиент-серверные архитектуры (CGI, PHP, ASP, RMI, EJB) [3];

– архитектуры, основанные на очередях сообщений (IBM MQ, ActiveMQ, RabbitMQ, Apache Kafka) [4, 5];

– GRID-архитектуры [6, 7].

Также для решения этой проблемы разработаны специализированные программные средства организации вычислительного процесса. В настоящее время набор используемых возможностей программных средств крайне ограничен ввиду отсутствия необходимых предоставляемых механизмов или их несовершенства. Эти средства удовлетворяют минимальные потребности функционирования комплекса, но никак не решают задачи интеграции, отладки и тестирования КФП.

В ряде комплексов в качестве технологии взаимодействия разработчики используют сокеты, что также не упрощает создание крупных информационно-управляющих систем. Необходимы дополнительные инструментальные средства, которые позволили бы решить задачи совместимости механизмов сетевого взаимодействия, форматов сообщений, логики взаимодействия, временной синхронизации, корректности передаваемых данных, защищенности от возможных ошибок в данных. На выявление причин возможных ошибок и недочетов уходят дни, недели и усилия целых подразделений разработчиков и системных инженеров.

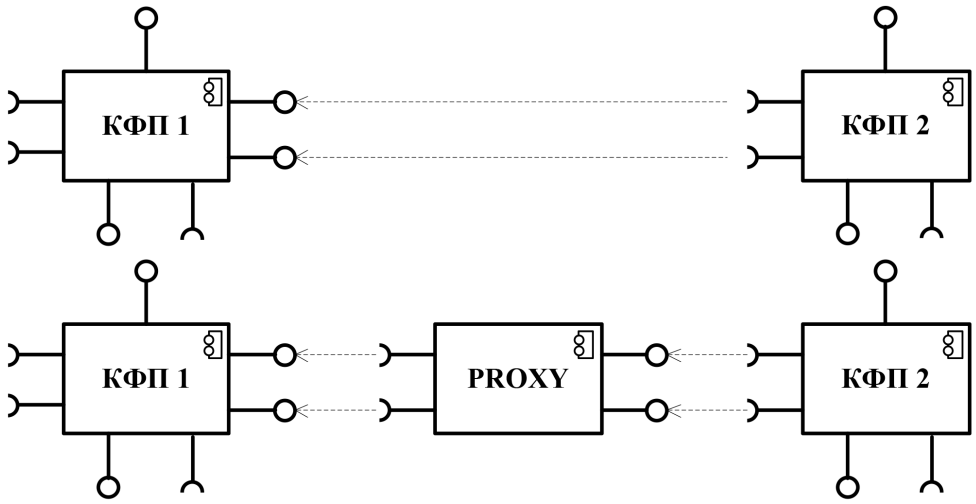

Схема взаимодействия КФП

Решение задачи

Одним из подходов, обеспечивающих наибольшее количество возможностей разработчикам при максимальной простоте, является использование программ-заместителей, реализующих паттерн проектирования «Proxy» [8, с. 203–211]. Программы-заместители устанавливаются в «разрыв» соединений между КФП (рисунок) и являются универсальными в следующих аспектах:

– заместители не зависят от специфики конкретной пары взаимодействующих программ;

– будучи единожды запущены и настроены, они могут использоваться для решения целого ряда задач.

Внедрение программ-заместителей в автоматизированные системы структурных подразделений компаний предоставляет новые возможности повышения эффективности управления сетевыми ресурсами. Позволяет упрощать процессы интеграции и отладки компонентов системы и повышает их надежность и производительность за счет мониторинга и корректировки сетевого трафика.

Альтернативным способом использования программ-заместителей является перехват сообщений взаимодействующих компонентов на уровне сетевых карт. Такой способ применяется программами-снифферами и реализуется с помощью библиотек захвата и анализа сетевых пакетов (packet capture library, pcap) [9]. Такой способ является технически более сложным и ограничивает набор возможных функций программ-заместителей, но позволяет избежать дополнительных действий по конфигурированию информационной системы для вставки в необходимые взаимодействия заместителей.

Программы-заместители могут предоставлять следующие возможности разработчикам:

1. Мониторинг корректности и состояния соединений.

2. Мониторинг информационных взаимодействий с просмотром передаваемых данных.

3. Регистрация (сохранение) передаваемых данных в файл.

4. Воспроизведение ранее зарегистрированных данных для воспроизведения картины входных данных.

5. Проверка корректности передаваемых данных.

6. Корректировка данных для устранения выявленных ошибок, имитации или нейтрализации данных, которые могут приводить к ошибкам.

7. Преобразование несовместимых форматов сообщений для интеграции различных версий компонентов или компонентов, использование которых ранее не было предусмотрено.

8. Отложенный запуск КФП, позволяющий отложить загрузку и инициализацию КФП до получения первого обращения к ним с целью минимизировать загрузку аппаратных и коммуникационных ресурсов.

9. Мониторинг общего объема и других статистических характеристик сетевых взаимодействий.

10. Проверка доступности взаимодействующих узлов и КФП.

11. Тестирование КФП корректными входными данными.

12. Тестирование КФП некорректными входными данными.

13. Нагрузочное тестирование вычислительной сети, отдельных КФП и информационной системы в целом.

14. Измерение пропускной способности каналов взаимодействия, отдельных КФП и информационной системы в целом.

15. Сбор данных для моделирования вычислительной сети.

16. Реконфигурирование (в том числе динамическое) информационной системы.

Несмотря на очевидную простоту реализации программ-заместителей, при их использовании возникает необходимость решения некоторых задач:

1. Распределение программ-заместителей по аппаратным ресурсам. Дополнительные функции обработки, выполняемые такими программами, расходуют аппаратные и коммуникационные ресурсы информационной системы. Как правило, дополнительные ресурсы для этих целей заранее не предусматриваются ввиду высокой стоимости оборудования, на котором может штатно функционировать информационная система. Конфигурация стендов, предназначенных для интеграции и тестирования, может располагать такими ресурсами, но они могут быть неравномерно распределены между узлами вычислительной сети. В этом случае требуется найти баланс между необходимыми в каждой конкретной конфигурации функциями программ-посредников и имеющимися запасами аппаратных и коммуникационных ресурсов для их использования.

2. Реализация алгоритмов обработки и генерации передаваемых данных. Такие алгоритмы требуют наличия достаточно полной информации о передаваемых между КФП сообщениях: их структуре, форматах передачи и представления данных, допустимых диапазонах значений, способах генерации тестовых данных (корректных и некорректных), в том числе отдельных значений и целостных сообщений.

При выборе варианта размещения программ-заместителей необходимо также учитывать следующие дополнительные аргументы:

– исключение дублирования Proxy на одном канале взаимодействия;

– расположение заместителя на стороне сервера при наличии нескольких подключающихся к нему клиентов позволяет избежать дублирования заместителей, которое неизбежно возникает при размещении заместителей на стороне каждого из клиентов;

– минимизация количества узлов, ресурсы которых используются для функционирования программ-заместителей, позволяет минимизировать отклонения от штатной конфигурации информационной системы и, как следствие, возможных отклонений в работе КФП;

– минимизация использования ресурсов каждого из узлов, которая также снижает вероятность отклонений в функционировании КФП на этих узлах;

– сокращение количества экземпляров программ-заместителей, так как при его увеличении возрастают накладные расходы на синхронизацию времени и генерируемых ими событий.

Задача описания структуры и содержания передаваемых данных также имеет множество решений. В информационных системах, создаваемых на большинстве предприятий, сообщения, как правило, описываются в виде структур данных на языке C/C++. Существует несколько способов для использования таких описаний:

– разработка специализированного средства разбора описаний структур на языке C/C++, поддерживающих используемые в них синтаксические конструкции;

– использование существующих средств разбора описаний. В качестве примера такого средства можно использовать соответствующий модуль системы автоматизированного формирования программной документации Doxygen;

– формирование дополнительных описаний на языке C/C++, упрощающих доступ к уже имеющимся структурам сообщений. Такой вариант уже реализован и эффективно используется в ряде технологических программ. Дополнительные описания позволяют не загромождать основные описания структур лишними инструкциями, в то же время они реализованы с использованием универсальных макросов, которые позволяют сделать дополнительные описания максимально короткими и простыми;

– разработка программного модуля (например, динамически подключаемого), обеспечивающего выдачу данных о структуре сообщений в необходимом формате;

– разработка дополнительных описаний структур на другом языке, например XML. Такой способ имеет недостаток, связанный с возможным несоответствием описаний на разных языках вследствие того, что упустить особенности компиляции в конкретной среде разработки достаточно просто (выравнивание полей структуры, разрядность целевой операционной системы, директивы прекомпиляции и др.).

Основной вариант получения описания структур подсказал опыт разработчиков библиотеки Boost, которая содержит

достаточно удобный программный интерфейс для описания сериализуемых структур данных [10]. Аналогичный интерфейс должен быть реализован для формирования и доступа к описаниям структур в программах-заместителях.

Проектирование и программная реализация программ-заместителей показала необходимость решения также и других проблем, в частности:

– разработка архитектуры, обеспечивающей возможность адаптации посредника к различным механизмам сетевого обмена между КФП. Используемые механизмы обмена могут различаться даже в рамках одной информационной системы;

– разработка алгоритмов представления данных для полей, имеющих специфический формат или смысл, который не описан на языке C/C++;

– разработка алгоритмов тестирования корректными и некорректными данными на основе описания структур сообщений [11]. Важно отметить, что сообщения не содержат полной информации о регламенте взаимодействия компонентов и др.

Указанные задачи выходят за рамки данной публикации и будут описаны в дальнейшем. Тем не менее предложенный механизм универсальных программ-заместителей позволяет с минимальными затратами решать достаточно широкий круг задач интеграции, отладки и тестирования КФП.

В процессе исследования рассмотрена технология, направленная на упрощение и автоматизацию процессов интеграции и тестирования компонентов информационно-управляющих систем. Применение при интеграции специальных программ-заместителей позволяет облегчить взаимодействие между различными элементами системы, выявлять и устранять ошибки, проводить анализ передаваемых данных.

Заключение

Применение программ-заместителей демонстрирует значительные преимущества в части мониторинга, регистрации и модификации сетевого трафика, что позволяет не только выявлять несоответствия и ошибки в работе компонентов, но и тестировать их работоспособность в разнообразных условиях. Такой подход обеспечивает более высокую гибкость и адаптивность системы в процессе ее разработки и эксплуатации, способствует сокращению времени на отладку и интеграцию, а также повышает общую надежность и производительность информационно-управляющих систем.

Список литературы

1. Асратян Р.Э., Лебедев В.Н. Интернет-служба конвейерной обработки защищенных информационных запросов в мультисетевой среде // Информационные технологии. 2017. Т. 23, № 8. С. 589–597.

2. Исикдаг У. Шаблоны проектирования для сервисно-ориентированных архитектур на основе BIM-технологий // Вестник ТГАСУ. 2013. № 2 (39). С. 185–212.

3. Хордов В.С., Игонин А.Г. Технологии распределенного проектирования // Вестник Ульяновского государственного технического университета. 2014. № 1 (65). С. 54–59.

4. Селезнёв С.П., Яковлев В.В. Архитектура промышленных приложений IoT и протоколы AMQP, MQTT, JMS, REST, CoAP, XMPP, DDS // International Journal of Open Information Technologies. 2019. Т. 7, № 5. С. 17–28.

5. Mateusz Kaczor, Paweł Powroźnik. Comparative analysis of message brokers // Journal Computer Sciences Institute. 2022. Is. 23. P. 89–96. URL: https://www.semanticscholar.org/reader/655e51cdf489f0d50be787f0c3884dc13c7fcaf6 (дата обращения: 24.09.2024).

6. Каменщиков М.А. Сервисы GRID как объекты стандартизации // Журнал радиоэлектроники. 2003. № 12. URL: http://jre.cplire.ru/jre/dec03/4/text.html (дата обращения: 30.03.2024).

7. Рябов Ю.Ф., Кирьянов А.К. Технология Грид // Школьные технологии. 2010. № 5. С. 134–145.

8. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Э. Гамма и др.; пер. с англ. А. Слинкина. СПб.: Питер, 2015. 368 с.

9. Рубцов В.В., Чичкин В.А., Мочинов А.А., Семенов М.Д. Генератор трафика. Создание пакетов с соединением TCP на примере программы SCAPY // Перспективы науки. 2021. № 7 (142). С. 18–22.

10. Лопин И.А., Никехин А.А. Применение библиотеки BOOST.COMPUTE для моделирования электростатического поля методом случайного блуждания // Объектные системы. 2016. № 13. С. 60–66.

11. Морозова В.И. Параллелизм в C++ на примере библиотеки Pthread // Молодой ученый. 2022. № 19 (414). С. 24–26.

12. Панков Д.А., Денисова Л.А. Контроль и диагностика неисправностей программно-аппаратного комплекса // Омский научный вестник. 2018. № 2 (158). С. 128–133.