

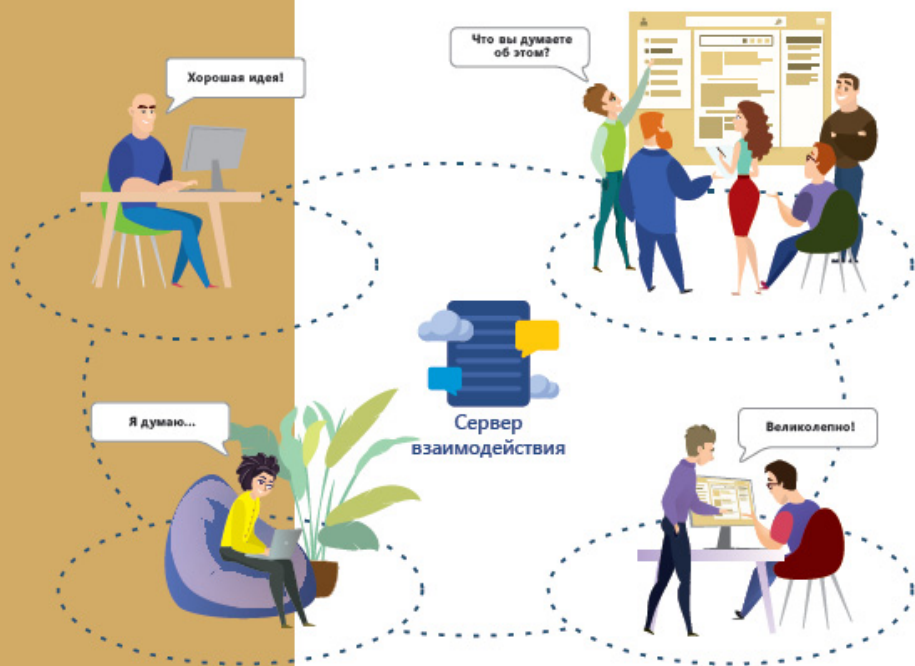


БИБЛИОТЕКА РАЗРАБОТЧИКА

Е. Ю. Хрусталева

СИСТЕМА ВЗАИМОДЕЙСТВИЯ

Коммуникации в бизнес-приложениях



РАЗРАБОТКА В СИСТЕМЕ
«1С:ПРЕДПРИЯТИЕ 8.3»



Хрусталева Е.Ю.

Система взаимодействия.

Коммуникации в бизнес-приложениях. Разработка в системе "1С:Предприятие 8.3"

Электронная книга в формате pdf, ISBN 978-5-9677-2877-8.

Электронный аналог издания "Система взаимодействия. Коммуникации в бизнес приложениях. Разработка в системе 1С:Предприятие 8.3" (ISBN 978-5-9677-2869-3, М.: ООО "1С-Паблишинг", 2019; артикул печатной книги по прайс-листу фирмы "1С": 4601546139733; по вопросам приобретения печатных изданий издательства "1С-Паблишинг" обращайтесь к партнеру "1С", обслуживающему вашу организацию, или к другим партнерам фирмы "1С").

Книга адресована специалистам, имеющим опыт разработки на платформе «1С:Предприятие». С ее помощью можно освоить новый платформенный механизм – система взаимодействия. Система взаимодействия позволяет реализовать живое общение прикладного решения с пользователями, а также пользователей одного или разных прикладных решений между собой. Причем такая коммуникация может быть привязана к выполнению как конкретных бизнес-задач, так и к обсуждению конкретных объектов, работа с которыми ведется в решении. Основные возможности этого механизма можно использовать без программирования, более сложные задачи реализуются посредством разработки с помощью встроенного языка "1С:Предприятия". В книге рассмотрены разные сценарии применения системы взаимодействия.

Один из примеров описывает взаимодействие пользователей друг с другом, которое можно выполнять «из коробки» без программных доработок.

Другой пример показывает, как организовать взаимодействие пользователей по определенному алгоритму, который описывается на встроенном языке.

Третий пример демонстрирует разработку "кадрового помощника" – работа, который отвечает на вопросы пользователей.

Также в книге рассматриваются два примера, в которых система взаимодействия используется для осуществления коммуникации по инициативе серверной части «1С:Предприятия». Первый пример – программирование алгоритма, реализующего функцию оповещения клиентского приложения о ходе выполнения длительной операции на сервере. Второй – программное формирование оповещений сотрудников об изменении состояния информации в базе данных (например, о том, что появился новый заказ, который нужно обработать и доставить клиенту). Для создания демонстрационных примеров использована версия 8.3.14.1411 платформы «1С:Предприятие 8».

Книга выпущена под редакцией Максима Радченко.

Дополнительные материалы

Информационные базы с примерами, описанными в книге, опубликованы на портале 1С:ИТС. Вы можете скачать их по адресу http://its.1c.ru/download/book_demo/.

Интернет-конференция для начинающих разработчиков
<http://devtrainingforum.v8.1c.ru/forum>.

Оглавление

5	Глава 1. Что такое система взаимодействия
35	Глава 2. Подключение и настройка
55	Глава 3. Примеры использования и сценарии применения системы взаимодействия
109	Глава 4. Программное взаимодействие

ГЛАВА 1

ЧТО ТАКОЕ СИСТЕМА ВЗАИМОДЕЙСТВИЯ

Назначение

Система взаимодействия – это механизм, позволяющий взаимодействовать между собой клиентским приложениям, серверу и пользователям одной или нескольких информационных баз.

Пользователи приложений, подключенных к системе взаимодействия, могут общаться между собой интерактивно, в режиме реального времени, с помощью сообщений, аудио- и видеозвонков. Сообщения существуют только в рамках некоторого обсуждения.

То есть нельзя «просто так» отправить сообщение какому-нибудь адресату. Обсуждения могут быть двух видов.

Во-первых, это могут быть обсуждения, не связанные с данными прикладного решения, посвященные какому-либо вопросу: мероприятиям, связанным с открытием нового магазина, вариантам проведения новогоднего праздника или грядущему совещанию у директора (рис. 1.1).

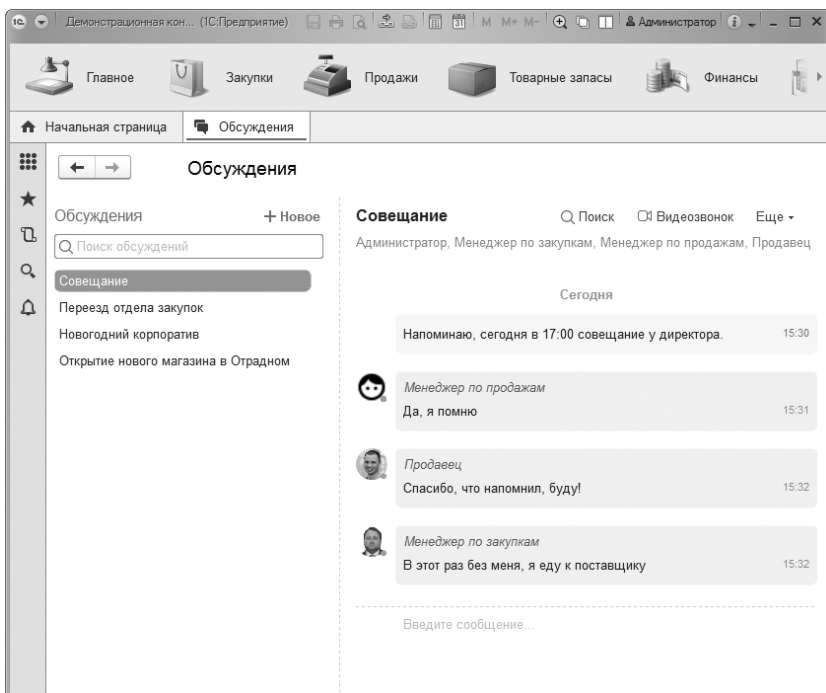


Рис. 1.1. Тематическое обсуждение пользователей

Во-вторых, это могут быть обсуждения, связанные с конкретными объектами данных: накладными, товарами, поставщиками и т. д. При этом вся переписка отображается в форме этих данных (рис. 1.2).

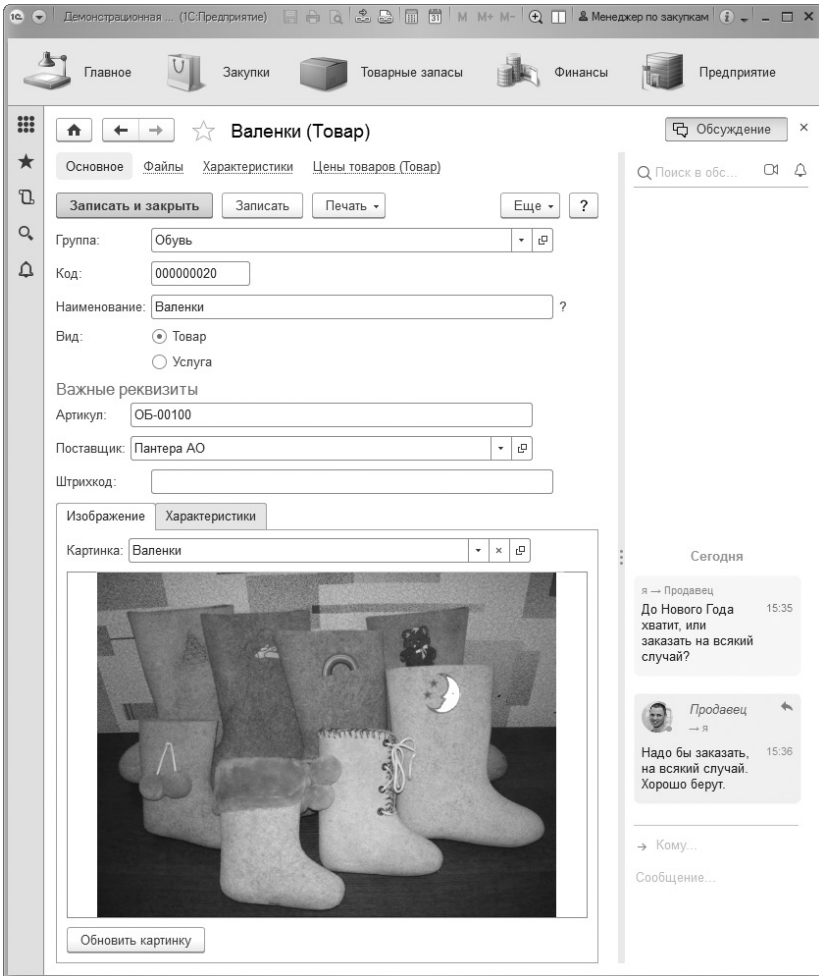


Рис. 1.2. Обсуждение пользователей, связанное с конкретным объектом данных

При наличии микрофона можно также звонить другим пользователям и общаться с ними голосом. А при наличии веб-камер пользователи могут видеть друг друга (рис. 1.3).

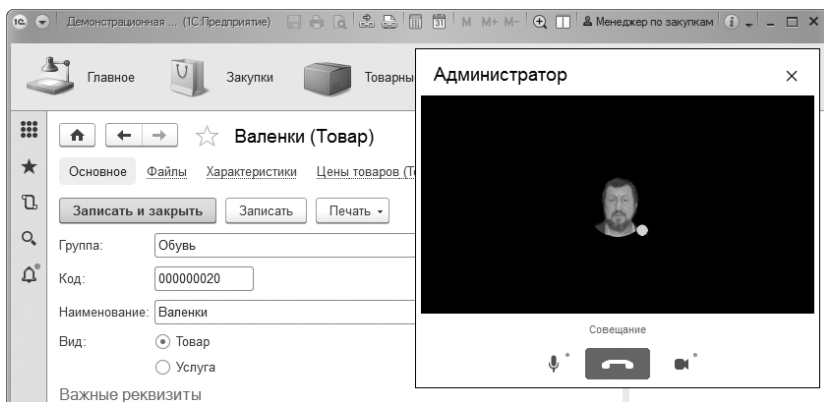


Рис. 1.3. Видеозвонок пользователя, у которого отсутствует веб-камера

Важно понимать, что система взаимодействия – это не какой-то инструмент абстрактного онлайн-общения наподобие Skype, WhatsApp и т.п., а средство коммуникации пользователей в контексте того приложения, с которым они работают. По этой причине система взаимодействия имеет как привычные функции коммуникации, так и специальные возможности интеграции с прикладными решениями.

Например, с помощью системы взаимодействия, ни на что не отвлекаясь и не покидая «1С:Предприятия», можно как решить все рабочие вопросы, связанные как с конкретным объектом информационной базы (например, с заказом покупателя), так и пообщаться на общие темы (например, что подарить на день рождения начальнику).

Кроме того, возможно также программное взаимодействие одного прикладного решения с другим прикладным решением и прикладного решения с человеком.

Также имеется возможность объединить несколько приложений в единое пространство взаимодействия. Например, менеджер, работающий в торговой программе, и бухгалтер, работающий

в бухгалтерской программе. Каждый из них может взаимодействовать с другими менеджерами или бухгалтерами, работающими в той же программе. И в то же время менеджер и бухгалтер могут взаимодействовать между собой (например, по поводу недостающих накладных на товары, договоров с поставщиками и т.п.), несмотря на то что они работают в разных программах.

Таким образом, система взаимодействия – это не просто «модный бантик» для пользователей, а полезный инструмент для разработчика, с помощью которого можно либо полностью автоматизировать, либо значительно ускорить выполнение отдельных этапов бизнес-процессов предприятия.

Устройство

Система взаимодействия реализована в клиент-серверной архитектуре. Она состоит из двух частей. Клиентская часть реализована непосредственно в платформе «1С:Предприятие 8». Серверная часть представляет собой отдельный программный продукт «1С:Предприятие – сервер взаимодействия», который может быть развернут в Интернете или в локальной сети организации.

Для удобства использования системы взаимодействия в Интернете (по адресу 1cDialog.com) на аппаратных мощностях фирмы «1С» развернут экземпляр этого сервера, носящий название «1С:Диалог». Это публичный сервер, которым могут пользоваться все желающие (рис. 1.4).

Работа системы взаимодействия построена на клиент-серверном обмене данными между прикладным решением системы «1С:Предприятие 8» и сервером системы взаимодействия. В качестве клиента сервера системы взаимодействия может выступать как одно из клиентских приложений системы «1С:Предприятие 8», так и серверная часть прикладного решения.



Рис. 1.4. Архитектура системы взаимодействия

Обмен сообщениями между клиентской частью и сервером взаимодействия осуществляется по протоколу WebSocket. Этот протокол обеспечивает защищенную передачу данных, что позволяет безопасно передавать сообщения, с помощью которых общаются пользователи, как в локальной сети, так и через Интернет.

Содержимое обсуждений, с помощью которых осуществляется взаимодействие пользователей, хранится на сервере системы взаимодействия.

Чтобы система взаимодействия стала доступной для использования в прикладном решении, информационную базу этого прикладного решения нужно зарегистрировать на сервере взаимодействия.

ПОДРОБНЕЕ

О регистрации приложений на сервере взаимодействия рассказывается во второй главе, в разделе «Регистрация приложений на сервере взаимодействия» на стр. 36.

Варианты использования

Система взаимодействия может использоваться как для интерактивного общения между живыми людьми, так и для программного взаимодействия одного прикладного решения с другим прикладным решением и прикладного решения с человеком.

При *интерактивном взаимодействии* все его участники в режиме реального времени могут обсуждать интересующие их вопросы, например возможность предоставления скидки контрагенту (рис. 1.5).



Рис. 1.5. Интерактивное взаимодействие

Смешанное взаимодействие выполняется между человеком с одной стороны и алгоритмом прикладного решения – с другой.

ПОДРОБНЕЕ

Об интерактивном взаимодействии рассказывается в разделе «Интерактивное взаимодействие» на стр. 16.

Простейшим примером такого взаимодействия является робот, отвечающий на определенные запросы (рис. 1.6).

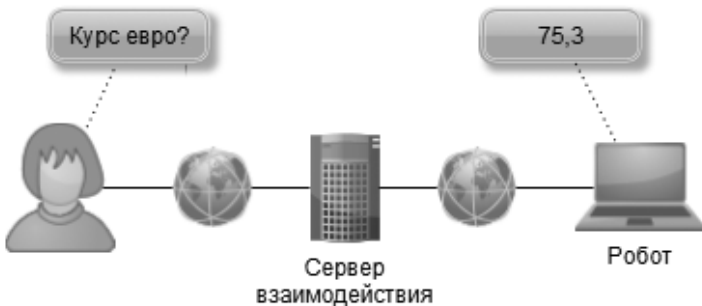


Рис. 1.6. Смешанное взаимодействие

При *программном взаимодействии* в обмене информацией принимают участие только процедуры прикладного решения. Как правило, через служебное обсуждение они оповещают друг друга о выполнении каких-то действий или наступлении каких-то событий. Например, сервер может посылать клиенту сообщение, что он закончил выполнять длительную операцию, которая была с этого клиента инициирована. Тем самым для клиента исчезает необходимость постоянно опрашивать сервер на эту тему (рис. 1.7).

ПОДРОБНЕЕ

О смешанном взаимодействии рассказывается в третьей главе, в разделе «Кадровый помощник (бот)» на стр. 84.

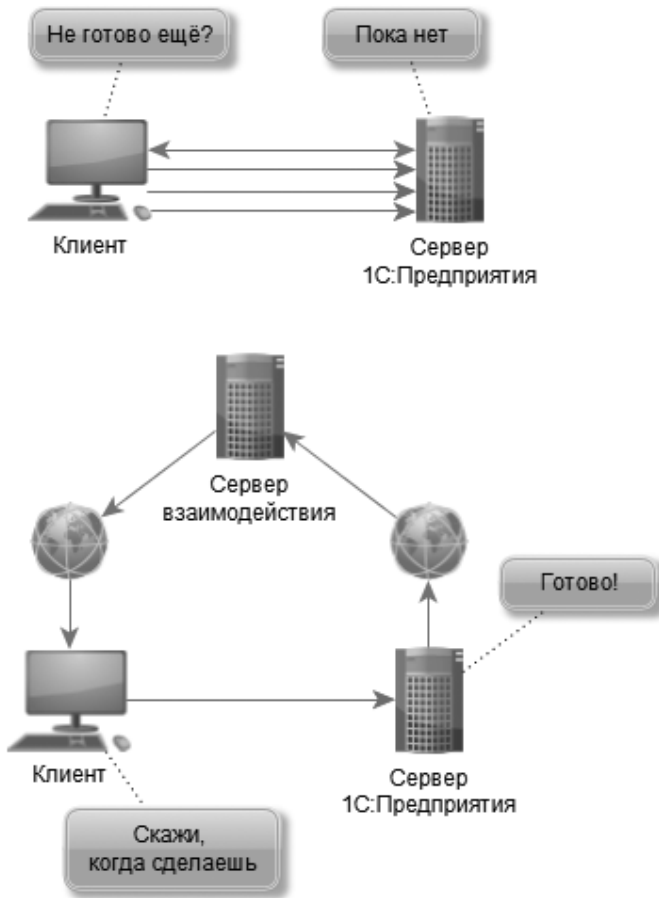


Рис. 1.7. Программное взаимодействие

ПОДРОБНЕЕ

О программном взаимодействии рассказывается в четвертой главе «Программное взаимодействие» на стр. 109.

Возможности системы взаимодействия

После регистрации информационной базы в системе взаимодействия в интерфейсе прикладного решения автоматически становятся доступны элементы, с помощью которых пользователи могут взаимодействовать друг с другом. Возможность работы с системой взаимодействия реализована в интерфейсе Такси в тонком клиенте, веб-клиенте и в управляемом режиме работы толстого клиента.

Все интерфейсные элементы для работы пользователя с сообщениями появляются в приложении автоматически и не требуют никакого программирования. В следующем разделе мы познакомимся с возможностями интерактивного взаимодействия пользователей.

Доступ к обсуждениям реализован двумя разными способами, и это зависит от того, связано обсуждение с данными прикладного решения или нет.

Если обсуждение не связано с данными (групповое обсуждение, обсуждение «один на один»), то оно отображается в основной форме системы взаимодействия. Чтобы открыть эту форму, нужно нажать на закладку Обсуждения, которая находится в панели открытых (рис. 1.8).

Если интерфейс прикладного решения настроен таким образом, что панель открытых не отображается, тогда можно воспользоваться центром оповещений. Сначала нужно нажать на значок «колокольчик» в панели инструментов, чтобы попасть в центр оповещений, а затем нажать кнопку Обсуждения (рис. 1.9).

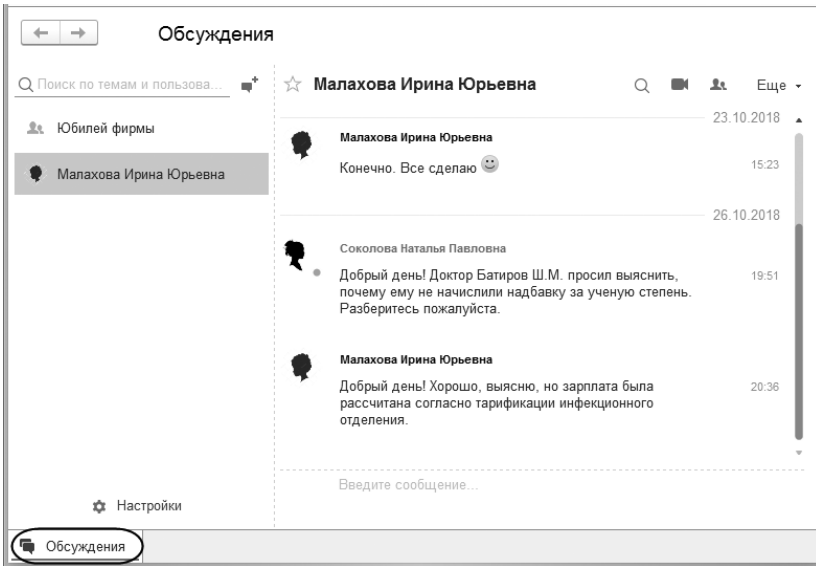


Рис. 1.8. Интерфейсные элементы системы взаимодействия

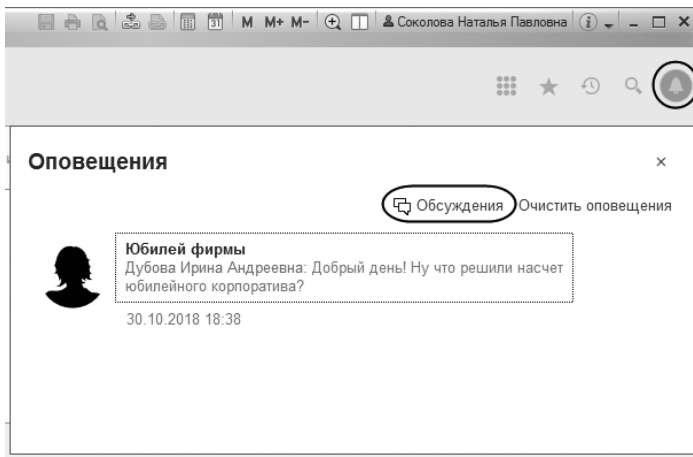


Рис. 1.9. Интерфейсные элементы системы взаимодействия

Обсуждения, которые связаны с данными прикладного решения (предметные обсуждения), привязаны к «своим» объектам данных и отображаются вместе с ними. Единого списка предметных обсуждений не существует, а каждый объект данных может иметь только одно предметное обсуждение.

Чтобы увидеть это обсуждение, нужно открыть форму объекта и нажать кнопку Обсуждение в заголовке формы рядом с кнопкой закрытия (рис. 1.10).

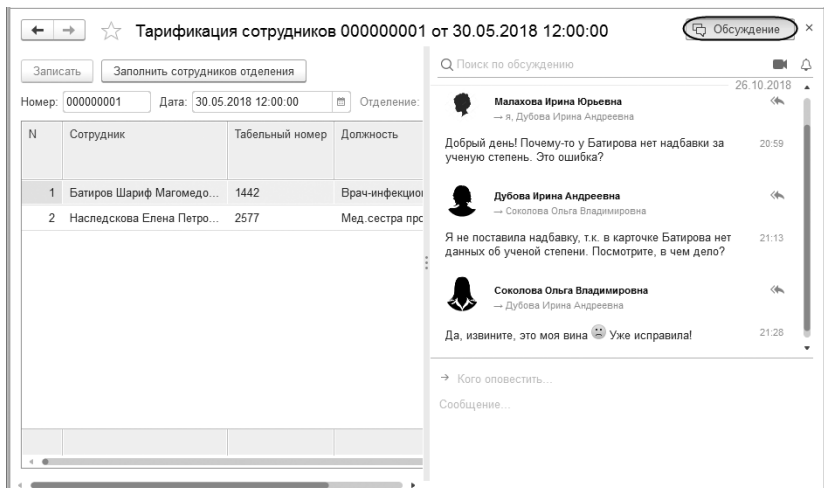


Рис. 1.10. Интерфейсные элементы системы взаимодействия


Интерактивное взаимодействие

Как уже говорилось, пользователи могут взаимодействовать друг с другом с помощью обсуждений. Обсуждения могут быть трех типов:

- *Групповое обсуждение*, не привязанное к объекту данных. Такое обсуждение может иметь произвольное количество участников (больше одного), которых может в любой момент добавлять любой участник обсуждения.

- *Обсуждение «один на один»*, не привязанное к объекту данных. Такое обсуждение имеет только двух участников, и для каждого из участников оно обозначается именем другого участника.
- *Предметное обсуждение*, привязанное к какому-либо объекту данных (например, документу, элементу справочника и т.п.). В таком обсуждении может принимать участие произвольное количество пользователей, но только тех, которые имеют право читать данные обсуждаемого объекта.

Групповое обсуждение

Каждый пользователь информационной базы может создать произвольное количество групповых обсуждений. Для этого он должен нажать на пиктограмму Новое обсуждение  в основной форме системы взаимодействия. Поскольку он является инициатором обсуждения, он автоматически становится его первым участником и затем может выбрать с помощью подбора других участников. Желательно также задать тему обсуждения, чтобы потом иметь возможность найти его в списке своих обсуждений (рис. 1.11).

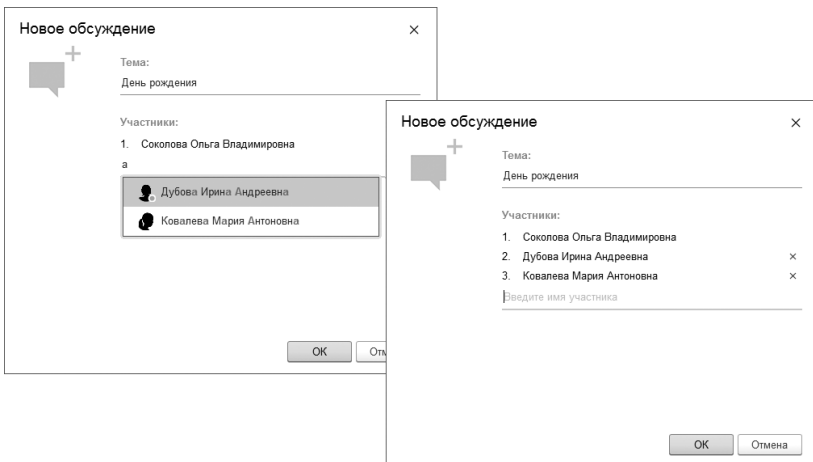


Рис. 1.11. Создание группового обсуждения

Например, кадровик может создать обсуждение, чтобы решить вопрос о подарке к дню рождению какого-то сотрудника. С помощью контекстного меню можно скопировать любое сообщение в обсуждении, а свое собственное сообщение пользователь может также отредактировать или удалить (рис. 1.12).

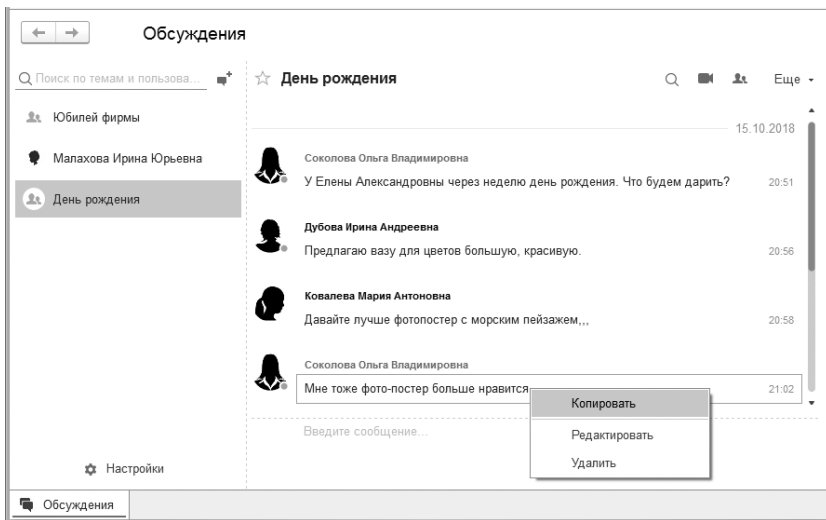


Рис. 1.12. Групповое обсуждение пользователей

В дальнейшем каждый участник группового обсуждения из меню Еще может изменить тему обсуждения, добавить в обсуждение новых участников или покинуть обсуждение. Эти действия также будут отражены в обсуждении (рис. 1.13).

Кроме того, можно получить ссылку на обсуждение, скопировать ее в буфер обмена и вставить эту ссылку, например, в другое обсуждение (см. рис. 1.14).

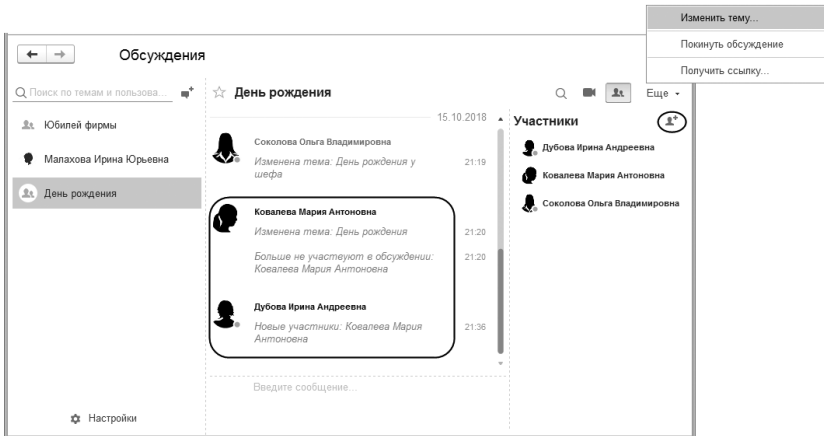


Рис. 1.13. Групповое обсуждение

Если пользователь решил покинуть обсуждение, оно автоматически удалится из списка его обсуждений. Самостоятельно присоединиться к покинутому обсуждению пользователь уже не сможет. Однако оставшиеся участники обсуждения могут снова добавить этого пользователя в число участников, и тогда это обсуждение снова появится в списке обсуждений этого пользователя.

Обсуждение «один на один»

Чтобы создать обсуждение «один на один», в основной форме системы взаимодействия в строку поиска (над списком обсуждений) нужно ввести фрагмент имени пользователя и выбрать подходящего собеседника из предложенного списка. В списке такое обсуждение будет обозначено именем второго участника. Для каждой пары пользователей такое обсуждение одно (рис. 1.14).

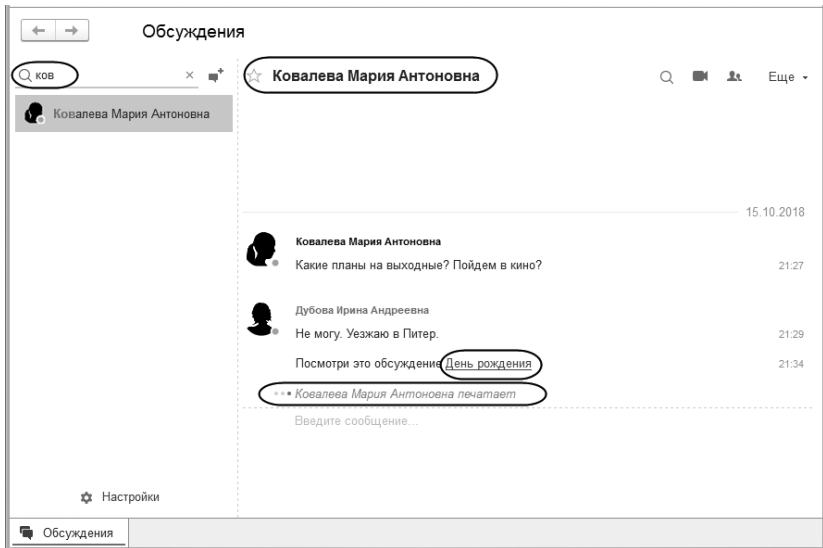


Рис. 1.14. Обсуждение пользователей «один на один»

Добавить новых участников и изменить тему обсуждения «один на один» нельзя. А также удалить такое обсуждение из списка обсуждений невозможно.

Отправка сообщений

Добавление сообщений в обсуждение и их отправка интуитивно понятны и не требуют пояснений.

Когда кто-то из пользователей печатает сообщение, остальные участники обсуждения видят служебное сообщение об этом (см. рис. 1.14).

Если пользователь хочет сделать свое сообщение более эмоциональным, он может вставить в текст смайлики с помощью стандартных наборов символов: «:)» – улыбка, «:(« – огорчение и т. п. (см. рис. 1.8, 1.10).

Отправляются сообщения либо по кнопке Отправить (с пиктограммой «самолетик»), либо по той клавише, которая задана в настройках системы взаимодействия. Окно настроек открывается по нажатию кнопки Настройки (с пиктограммой «шестеренка») внизу под списком обсуждений. Стандартно сообщения отправляются при нажатии клавиши Enter (рис. 1.15).

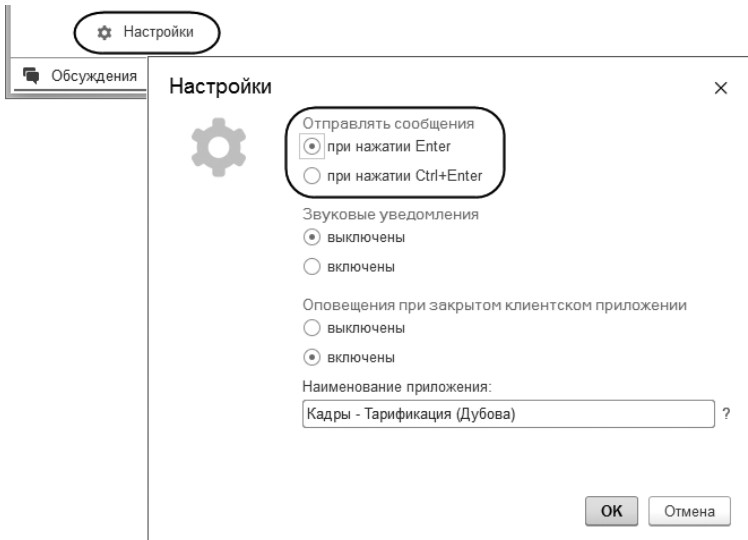


Рис. 1.15. Настройки системы взаимодействия

После отправки сообщения каждый участник обсуждения получит оповещение с текстом этого сообщения. Оно всплывет на 4 секунды в правом нижнем углу экрана, а затем попадет в центр оповещений. Нажав на это оповещение, пользователь попадет в соответствующее обсуждение, из которого он может написать и отправить свое сообщение (рис. 1.16).



Рис. 1.16. Добавление сообщений в обсуждение

Предметное обсуждение

Предметное обсуждение можно вести в формах элементов ссылочных типов и в формах записей регистров сведений, которые не блокируют окно владельца или весь интерфейс. В таких формах каждому пользователю, имеющему права чтения объекта, который хочется обсудить, становится доступна команда Обсуждение. С ее помощью открывается панель, в которой можно вести предметное обсуждение объекта информационной базы, отображаемого в форме. Такое обсуждение для каждого объекта одно. Единого списка предметных обсуждений не существует, и удалить такое обсуждение невозможно.

Например, экономист и кадровик могут уточнять процент надбавки за вредность у какой-то позиции штатного расписания (рис. 1.17).

Просматривать предметное обсуждение и участвовать в нем может любой пользователь информационной базы, имеющий права на чтение того объекта, который является предметом обсуждения.

Инфекционный кабинет (Штатное расписание)

Отделение: **Инфекционный кабинет** | Наименование: **Инфекционный кабинет**

N	Должность	Количество ставок	ПЮ/Мин		Средний оклад	Процент надбавки за вредность
			ПЮ/Макс	Тариф макс		
1	Врач-инфекционист	1,00	Врач-специалист б/к ср...	29 853,00	33 589,00	15
			Врач-специалист выс...	37 325,00		
2	Мед. сестра процедурная	1,00	Мед. сестра б/к средн	15 398,00	17 870,50	15
			Мед. сестра высшей к...	20 343,00		
3	Санитарка	1,00	Санитарка без стажа ...	11 881,00	12 179,00	15
			Санитарка без стажа ...	12 477,00		

Обсуждение

Поиск по обсуждению

Сегодня

Дубова Ирина Андреевна → Соколова Ольга Владимиро... 20:57
 Проверьте пожалуйста: у медсестры процедурной тоже должна быть указана вредность 15%. Иначе срока вакаций и тарификации неправильно считается

Соколова Ольга Владимиро... → 21:00
 Да, это ошибка. Сейчас исправлю

Исправила 21:01

Дубова Ирина Андреевна → Соколова Ольга Владимиро... 21:04
 Спасибо

→ Кого оповестить...
 Сообщения

Рис. 1.17. Предметное обсуждение

Если автор сообщения хочет оповестить конкретного пользователя (или пользователей), он может указать их в поле Кого оповестить. И тогда эти пользователи получат оповещение об этом сообщении. Как правило, обсуждения происходят в кругу постоянных участников, поэтому для ответа на сообщение удобно нажимать кнопки Ответить или Ответить всем (см. рис. 1.17).

Если пользователь хочет быть в курсе обсуждения какого-то объекта, ему нужно включить режим наблюдения в форме этого объекта (нажать на пиктограмму «колокольчик», см. рис. 1.17). И тогда он будет получать оповещения обо всех новых сообщениях в этом обсуждении, даже если этот пользователь не указан среди тех, кого автор сообщения хотел оповестить.

После щелчка по оповещению пользователь, которому адресовано сообщение, сразу попадет в форму обсуждаемого объекта и может написать ответ.

Как уже говорилось, команда Обсуждение и соответствующая кнопка в правом углу формы становится доступна в формах элементов ссылочных типов и в формах записей регистров сведений, которые не блокируют окно владельца или весь интер-

фейс. При этом свойство формы `ОтображениеОбсуждений` стандартно имеет значение `Авто` (рис. 1.18).

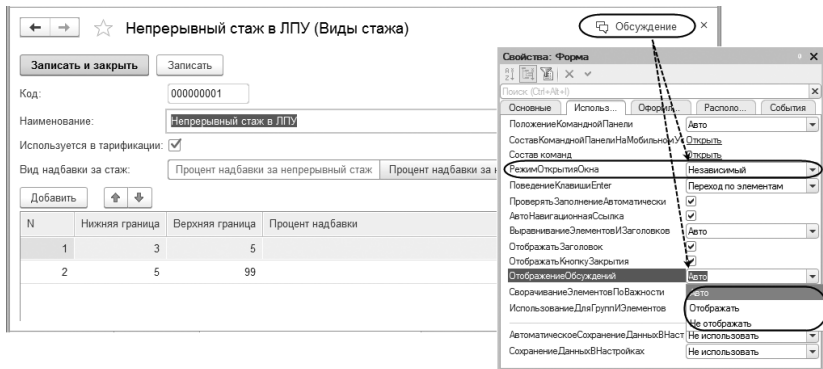


Рис. 1.18. Свойство формы «ОтображениеОбсуждений»

Но если в форме не нужно ничего обсуждать, то можно ради экономии места скрыть кнопку `Обсуждение` и саму панель обсуждений, которую эта кнопка открывает. Для этого можно в конфигураторе установить свойство формы `ОтображениеОбсуждений` в значение `Не отображать` или программно выбрать это значение из значений системного перечисления `ОтображениеОбсужденийФормы`.

Информация о пользователях

Пользователи информационной базы, подключенной к системе взаимодействия, могут задать свой аватар (картинку), телефонный номер и адрес электронной почты. Это делается в специальном диалоге, вызываемом нажатием на ссылку с именем пользователя в заголовке приложения (рис. 1.19).

Дополнительная информация о пользователях желательна, но вовсе не обязательна для работы системы взаимодействия.

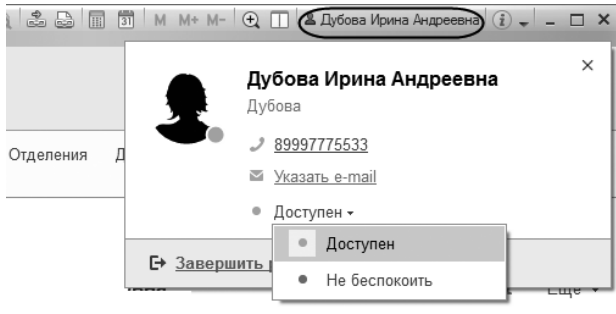


Рис. 1.19. Параметры пользователя

У пользователя, работающего с информационной базой, подключенной к системе взаимодействия, автоматически устанавливается статус **Доступен**, который отображается в виде зеленой точки рядом с аватаром. После пяти минут отсутствия его статус меняется на **Отшел** (отображается в виде желтой точки). Кроме того, он может изменить свой статус на **Не беспокоить** (отображается в виде красной точки), и тогда он не будет получать всплывающие оповещения о новых сообщениях и звонках, но позднее он может увидеть их в центре оповещений. Если пользователь еще не начинал или уже закончил работу с приложением, у него автоматически устанавливается статус **Не в сети** (отображается только аватар без каких-либо точек).

Поиск

Чтобы быстро перейти к нужному обсуждению в основной форме взаимодействия, можно воспользоваться строкой поиска над списком обсуждений (слева сверху, с пиктограммой «лупа»). Поиск проводится среди тем обсуждений (для групповых обсуждений) и имен пользователей (для обсуждений «один на один»). В результате выдаются все обсуждения, наименования которых содержат символы, набранные в строке поиска.

Если же нужно найти что-то в тексте сообщений, то можно воспользоваться строкой поиска над списком сообщений в обсуждении (справа сверху, с пиктограммой «лупа»). Поиск проводится только среди сообщений открытого в данный момент обсуждения (в т.ч. предметного). В результате цветом выделяются все найденные вхождения, и по ним можно передвигаться вперед/назад. Если символы, набранные в строке поиска, не найдены, то они выделяются красным цветом (рис. 1.20).

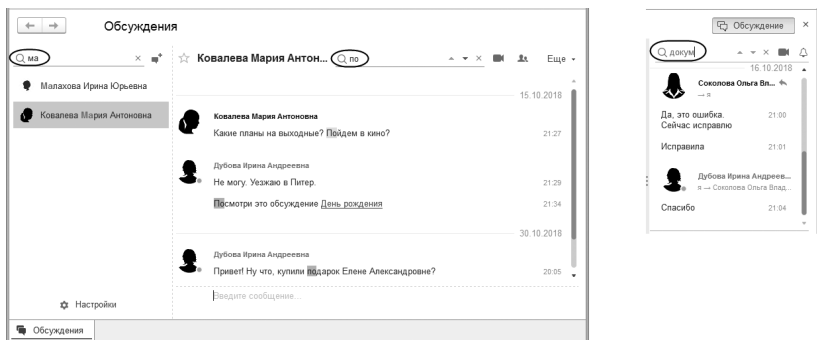


Рис. 1.20. Поиск в обсуждениях и тексте сообщений

Видеозвонки

Из любого обсуждения пользователь может позвонить другому (или сразу нескольким) пользователям. При наличии камер собеседники смогут еще и видеть друг друга.

Нажав на имя пользователя в списке сообщений в обсуждении, можно начать видеозвонок конкретному пользователю прямо из окна с дополнительной информацией о нем. Или же можно нажать кнопку Видеозвонок над списком сообщений в обсуждении и подобрать нужных собеседников (рис. 1.21).



Рис. 1.21. Видеозвонок

Если видеозвонок выполняется из обсуждения, то платформа автоматически предлагает выбрать собеседника из тех пользователей, которые в этом обсуждении участвуют. Однако вручную в качестве собеседника можно выбрать любого другого пользователя информационной базы.

В окне исходящего вызова отображаются аватар и имя вызываемого пользователя, а затем собственное изображение (при наличии веб-камеры). В окне входящего вызова показываются аватар и имя вызывающего пользователя с возможностью принять или отклонить вызов. Кроме того, показывается ссылка на обсуждение, из которого был сделан звонок, с помощью которой вызываемый собеседник может быстро перейти к обсуждению и ознакомиться с ним при необходимости (рис. 1.22).

Если вызываемый собеседник по каким-то причинам не смог ответить на звонок, то у него появится оповещение о пропущенном вызове (рис. 1.23).

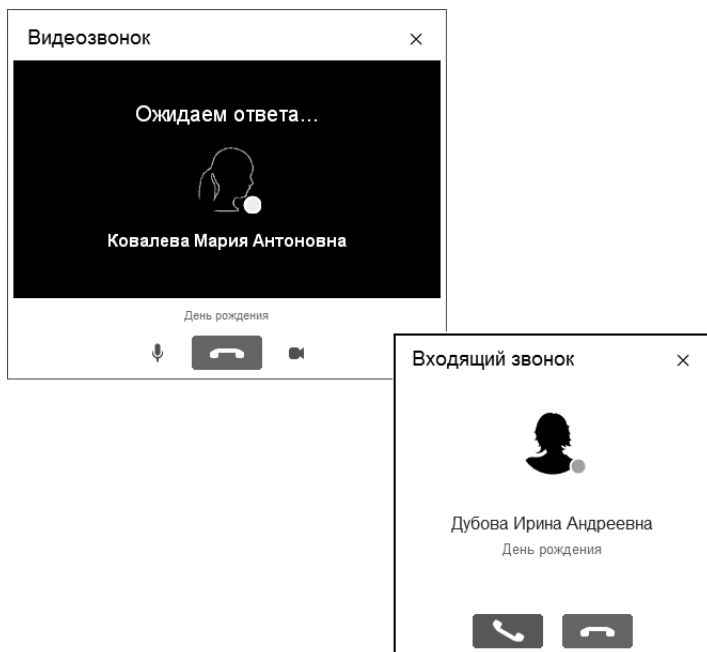


Рис. 1.22. Видеозвонок

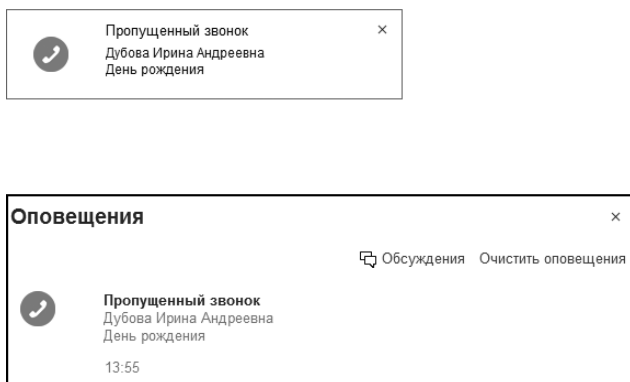


Рис. 1.23. Оповещение пользователя о пропущенном звонке

Помимо звонков конкретному пользователю можно также организовывать видеоконференции, то есть звонить сразу нескольким пользователям. При этом любой из участников видеозвонка может при необходимости добавлять или удалять других пользователей. Таким образом, можно быстро подключить к своему видеозвонку нового участника, чтобы уточнить у него обсуждаемый вопрос (рис. 1.24, 1.25).

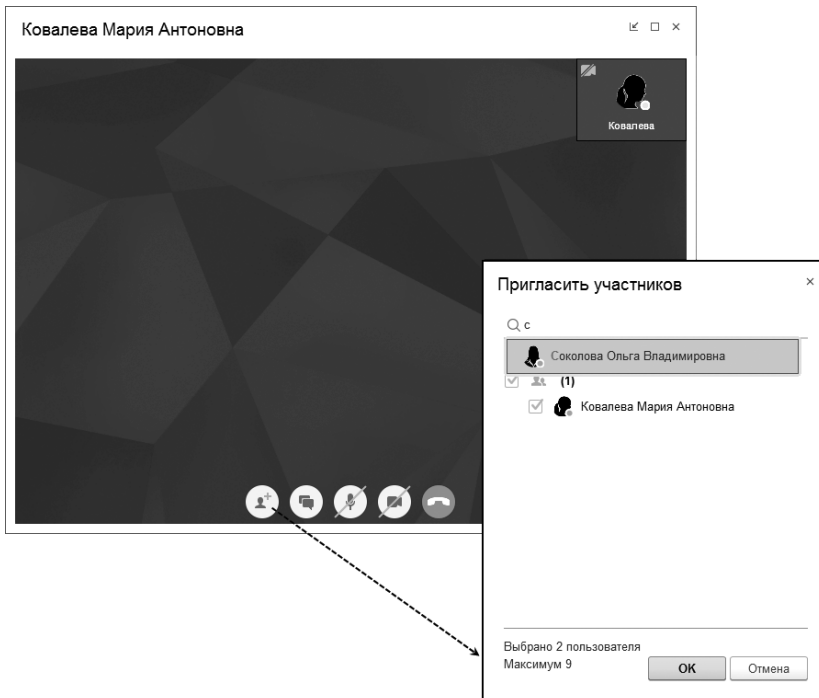


Рис. 1.24. Видеоконференция

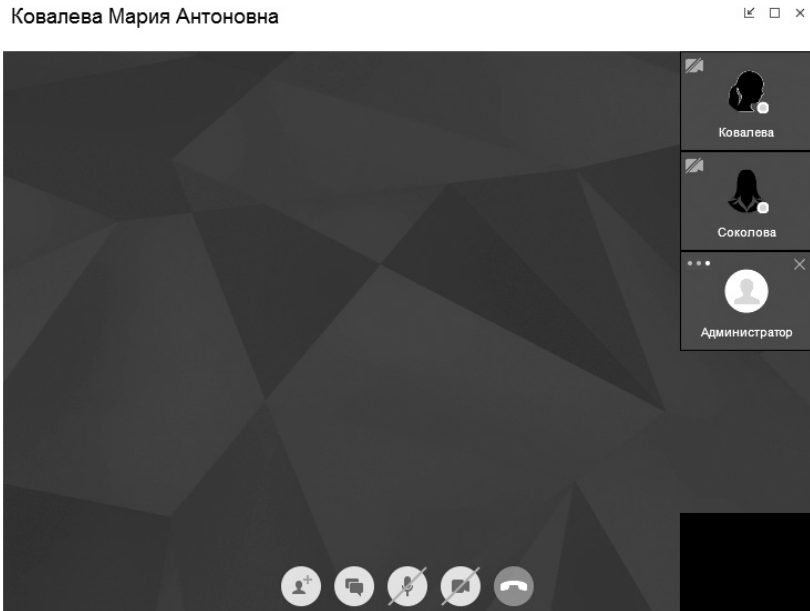


Рис. 1.25. Видеоконференция

На момент написания книги на сервере взаимодействия действует стандартное ограничение на количество участников – не более десяти. Но при использовании собственного сервера взаимодействия можно изменить это ограничение.

В режиме видеоконференции существует удобная возможность управлять расположением участников группового видеозвонка. Можно сделать так, что видео от подключенных участников будут занимать равные области окна, а неподключенные участники будут отображаться в правой части в виде плиток. Собственное видео при этом отображается в правой нижней части окна.

Кроме того, можно выделить одного из участников так, что его видео будет отображаться на все окно, а остальные – в виде плиток в правой части окна.

Передача файлов в сообщении

В версии платформы 8.3.14 появилась очень удобная и полезная возможность – вложение в сообщение произвольных файлов. Файлы, прикрепленные к сообщению, отображаются в виде гиперссылок после текста сообщения, до действий сообщения. У гиперссылки отображается размер файла, а также картинка, соответствующая типу файла (рис. 1.26).

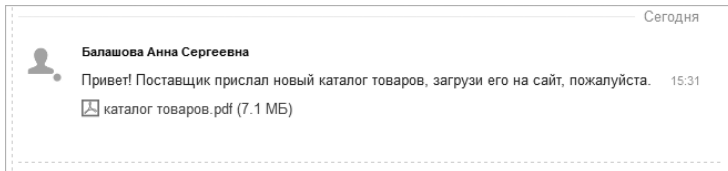


Рис. 1.26. Гиперссылка файла, вложенного в сообщение

Для прикрепления файлов к сообщению можно выполнить следующие действия:

- нажать на кнопку Прикрепить файл (значок в виде скрепки) в поле ввода сообщения (рис. 1.27);
- перетащить файл из операционной системы;
- вставить изображение (картинку) из буфера обмена.

При вложении картинки из буфера обмена открывается окно предварительного просмотра, в котором можно задать имя вставляемой картинки.

При отправке сообщения с вложенными файлами эти файлы помещаются по внешнее файловое хранилище на сервере взаимодействия. В то время, пока файлы загружаются в хранилище, в сообщении отображаются индикатор прогресса и общий процент загрузки на все файлы сообщения (рис. 1.28).

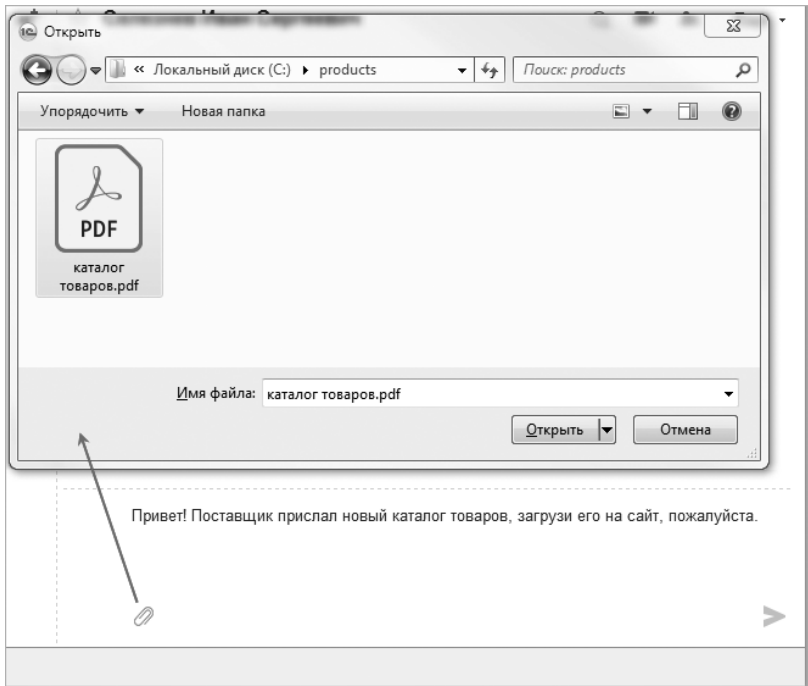


Рис. 1.27. Прикрепление файла к сообщению

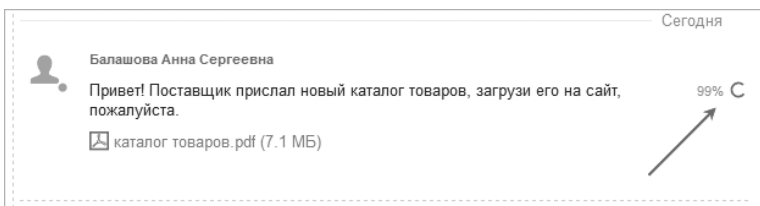


Рис. 1.28. Процесс загрузки файлов в хранилище при отправке сообщения

Файлы, прикрепленные к сообщениям, хранятся на сервере взаимодействия, во внешнем хранилище. Поддерживаются хранилища Amazon S3, Google Cloud, OpenStack Swift и другие хранилища, реализующие протокол Amazon S3.

Если на сервере взаимодействия внешнее хранилище не подключено, то в интерфейсе клиентских приложений будут отсутствовать возможности прикрепления файлов к сообщениям.

При использовании собственного сервера системы взаимодействия можно наложить ограничения на некоторые характеристики вложений. Например, можно ограничить:

- размер одного загружаемого файла,
- суммарный размер загружаемых файлов в сутки,
- суммарный размер получаемых файлов в сутки.

Если клиентское приложение пытается превысить одно из этих ограничений, то сообщение с таким вложенным файлом не отправится, а в подсказке значка, сигнализирующего об ошибке при отправке сообщения, отобразится текст, соответствующий нарушенному ограничению (рис. 1.29).

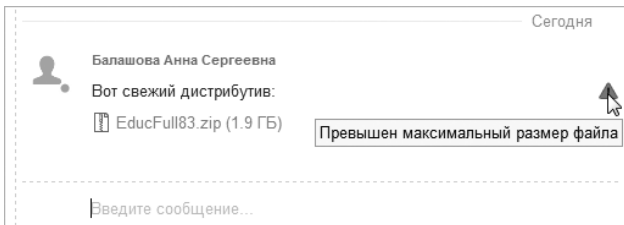


Рис. 1.29. Ошибка при отправке сообщения с файлом, превышающим допустимый размер вложения

Прикрепление файлов к сообщению можно реализовать также и помощью встроенного языка. Для этого используется программный объект КоллекцияВложенийСистемыВзаимодействия. Коллекция вложений имеется у каждого сообщения и доступна через новое свойство сообщения – Вложения. Она состоит из объектов ВложениеСистемыВзаимодействия.

ПРИМЕЧАНИЕ

Пример создания сообщения и прикрепления к нему файла показан в третьей главе, в разделе «Общение кадровика с сотрудниками в обсуждении "один на один" и в групповом обсуждении» (см. листинг 3.11).

ГЛАВА 2

ПОДКЛЮЧЕНИЕ И НАСТРОЙКА

Для того чтобы в приложении стала доступной работа с системой взаимодействия, нужно зарегистрировать это приложение (информационную базу) на сервере взаимодействия.

Если требуется настроить взаимодействие нескольких приложений между собой, то при регистрации приложений на сервере взаимодействия нужно включить эту возможность и параметры совместного использования приложений.

Для создания примеров, описываемых в книге, чтобы ничего не усложнять, мы просто будем подключаться к серверу взаимодействия, который установлен на инфраструктуре фирмы «1С» в виде публичного сервиса «1С:Диалог».

Но сервер взаимодействия существует в виде отдельного программного продукта, его можно установить самостоятельно в своей локальной сети и использовать аналогичным образом. Это может быть востребовано в том случае, если важно, чтобы конфиденциальная информация (в виде сообщений) не выходила за пределы локальной сети организации.

Для более удобной и оперативной работы с системой взаимодействия служит программа «1С:Предприятие – оповещения и запуск» (или агент клиентского приложения), которую нужно установить на компьютер.

Рассмотрим эти шаги более подробно.

Регистрация приложений на сервере взаимодействия

Группа приложений, которые используют систему взаимодействия, обозначается в сервисе взаимодействия термином *абонент*. Идентификатором абонента является адрес электронной почты, указанный при регистрации приложения в сервисе. Один абонент может зарегистрировать сразу несколько приложений.

Зарегистрировать приложение на сервере взаимодействия может пользователь информационной базы с правами *Регистрация Информационной Базы Системы Взаимодействия*. Тот, кто выполнил регистрацию приложения, становится *владельцем абонента* системы взаимодействия.

Регистрация приложения выполняется в режиме 1С:Предприятие с помощью стандартной обработки *Управление системой взаимодействия*. Эта обработка становится доступной только для тех информационных баз, в которых существуют пользователи (рис. 2.1).

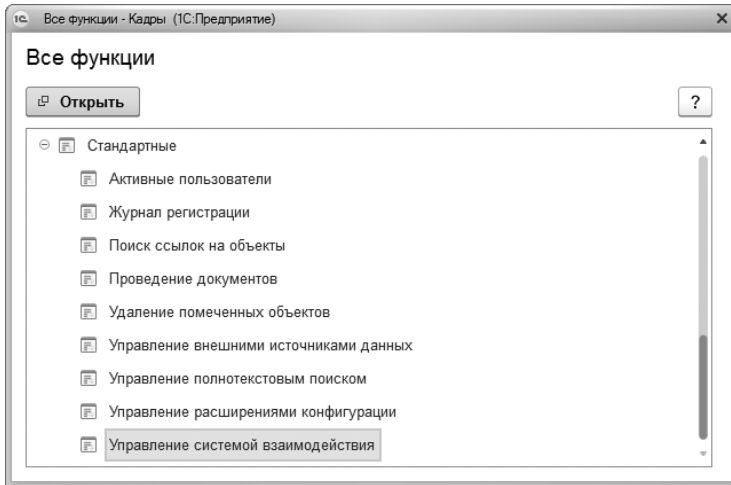


Рис. 2.1. Обработка «Управление системой взаимодействия»

При регистрации нужно указать электронный адрес, на который будет выслан код регистрации, и нажать кнопку Получить код. Затем ввести полученный код в поле Код регистрации и нажать кнопку Зарегистрировать (рис. 2.2).

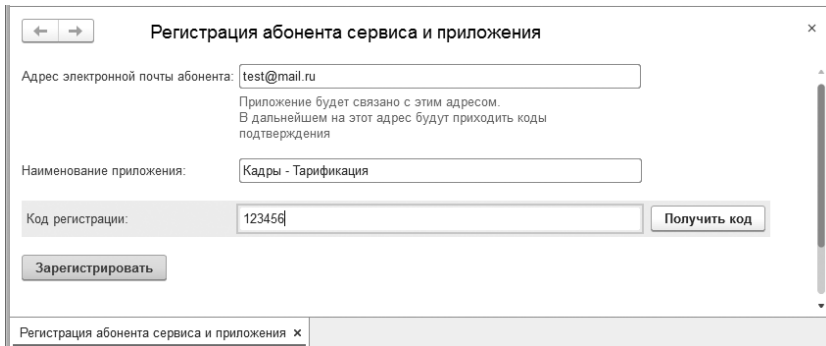


Рис. 2.2. Регистрация абонента сервиса и приложения

Поясним, что адрес электронной почты – это единственный уникальный идентификатор, по которому один абонент отличается от другого. Поэтому для того, чтобы вести реальную работу с системой взаимодействия, при регистрации нужно использовать доступный и хорошо знакомый электронный адрес.

Если в сервисе еще нет абонента с указанным адресом электронной почты, он создается и регистрируется автоматически. В дальнейшем на этот же адрес можно зарегистрировать другие приложения, которые также будут принадлежать этому абоненту.

После успешной регистрации приложения появляется окно, в котором можно настроить совместное использование всех зарегистрированных приложений этого абонента (об этом будет рассказано в следующем разделе) или отменить регистрацию приложения в системе взаимодействия (рис. 2.3).



Рис. 2.3. Отмена регистрации приложения в системе взаимодействия

После подтверждения отмены регистрации и нажатия кнопки 'Отменить регистрацию' работа приложения с системой взаимодействия станет невозможна. Соответствующие элементы интерфейса станут недоступны. Однако если вновь зарегистрировать приложение на этого же абонента (с тем же адресом электронной почты), то доступ к обсуждениям будет восстановлен.

Совместное использование приложений

Если у абонента (на предприятии) зарегистрировано несколько приложений, то можно объединить их в единое информационное пространство (с точки зрения системы взаимодействия). Тогда пользователи объединенных приложений смогут общаться между собой так же, как если бы они находились внутри одного приложения.

Например, в организации производится расчет зарплаты сотрудников на основе их тарификации. При этом у организации существуют два разных приложения «Кадры – Тарификация» и «Кадры – Зарплата».

После регистрации первого приложения в системе взаимодействия (см. рис. 2.2) его пользователи, например кадровик и экономист, могут обсуждать между собой конкретные кадровые вопросы (см. рис. 1.17) или какие-то общие темы.

После регистрации второго приложения в системе взаимодействия его пользователи также могут взаимодействовать между собой. Например, главный бухгалтер может напомнить расчетчику зарплаты о сроках предоставления расчетных ведомостей (рис. 2.4).

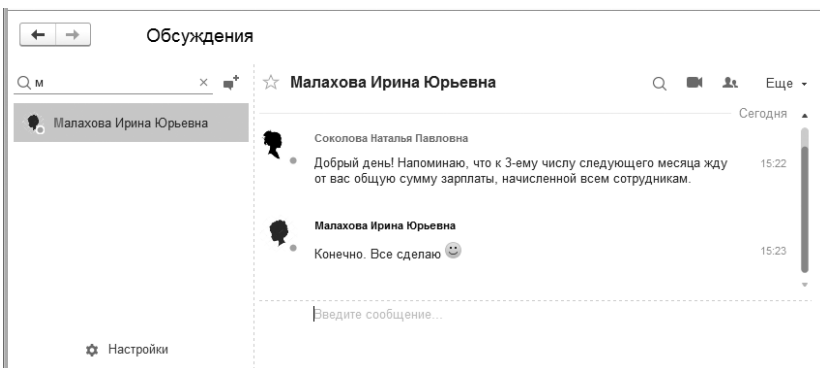


Рис. 2.4. Обсуждение «один на один» в приложении «Кадры – Зарплата»

Приложения «Кадры – Тарификация» и «Кадры – Зарплата» логически связаны между собой, и между ними настроен обмен данными. В частности, документы по тарификации сотрудников создаются и рассчитываются в приложении «Кадры – Тарификация» и передаются по плану обмена в приложение «Кадры – Зарплата». Там на основе этих документов начисляется и рассчитывается зарплата сотрудников.

Поэтому логично объединить эти приложения и для системы взаимодействия тоже, чтобы сотрудники обоих приложений могли обсуждать между собой различные рабочие вопросы. Для этого нужно настроить совместное использование этих приложений. Сделать это может только владелец абонента, то есть тот пользователь, который зарегистрировал приложения в системе взаимодействия.

Итак, откроем стандартную обработку Управление системой взаимодействия и выполним команду Совместное использование приложений. Затем над появившимся списком совместно используемых приложений нажмем кнопку Добавить и отметим те приложения, которые мы ранее уже зарегистрировали в системе взаимодействия и которые мы хотим объединить (рис. 2.5).

Кроме того, установим параметры Сопоставление пользователей и Сопоставление предметных обсуждений, которыми будет руководствоваться система взаимодействия при совместном использовании приложений.

Остановимся на первом параметре подробнее. Дело в том, что после объединения приложений в качестве пользователей системы взаимодействия выступают пользователи всех объединенных приложений.

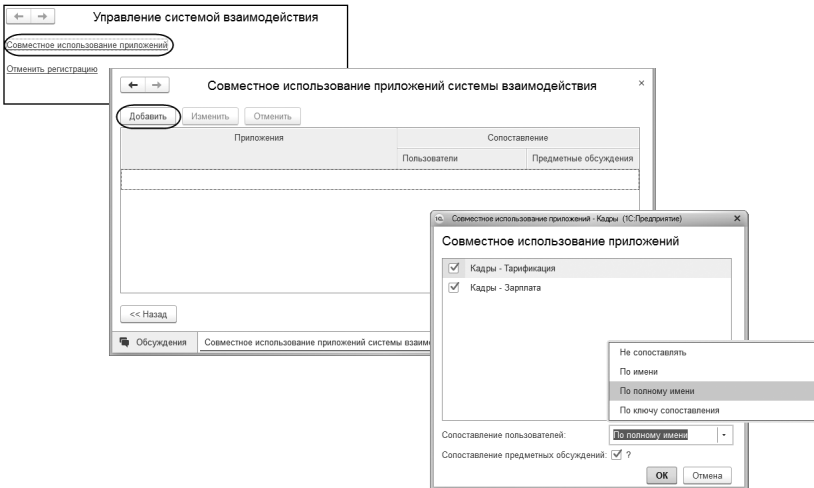


Рис. 2.5. Настройка совместно используемых приложений

Параметр Сопоставление пользователей позволяет указать, каким образом будут сопоставляться пользователи объединяемых приложений:

- По имени – в этом случае одинаковыми будут считаться пользователи системы взаимодействия, у которых одинаковое свойство Имя.
- По полному имени – в этом случае одинаковыми будут считаться пользователи системы взаимодействия, у которых одинаковое свойство ПолноеИмя.
- По ключу сопоставления – в этом случае одинаковыми будут считаться пользователи системы взаимодействия, у которых одинаковое свойство КлючСопоставления.
- Не сопоставлять – в этом случае все пользователи будут считаться различными пользователями системы взаимодействия.

Весьма вероятна ситуация, когда один и тот же человек может быть пользователем различных приложений. И если не сопоставлять пользователей объединяемых приложений, то для системы

взаимодействия этот человек будет считаться разными пользователями. Соответственно, появятся дубли при подборе пользователей в обсуждение и различные списки обсуждений пользователя в каждом приложении, хотя это один и тот же человек.

При сопоставлении пользователей по имени однофамильцы могут «пострадать», так как система взаимодействия будет считать всех этих людей одним пользователем. И в результате различные пользователи начнут видеть не «свои» обсуждения.

Например, в приложении «Кадры – Тарификация» есть кадровик Соколова Ольга Владимировна, в приложении «Кадры – Зарплата» есть бухгалтер Соколова Наталья Павловна, а имя у этих пользователей одно – «Соколова». Чтобы система взаимодействия не путала этих людей, лучше сопоставлять пользователей по полному имени.

Если среди пользователей объединяемых приложений есть полные тезки, тогда можно сопоставлять пользователей по ключу сопоставления, который задается программно в виде уникальной текстовой строки в специальной обработке (подробнее см. раздел «Рекомендуемые сценарии подключения» на стр. 48).

Ключ сопоставления пользователей может понадобиться не только для того, чтобы различать пользователей, но и, наоборот, чтобы их правильно сопоставлять при совместном использовании. Например, в одном приложении человек может работать как пользователь «Иванова Ольга Сергеевна», а в другом – как «Олечка». Чтобы система взаимодействия «поняла», что это один и тот же человек, у этих пользователей в обоих приложениях должен быть установлен одинаковый ключ сопоставления.

Итак, отметим приложения для совместного использования в системе взаимодействия, установим параметр Сопоставление пользователей в значение По полному имени, а также установим флажок у параметра Сопоставление предметных обсуждений (назначение этого параметра мы рассмотрим ниже). Нажмем ОК.

Список совместно используемых приложений системы взаимодействия примет следующий вид (рис. 2.6).



Рис. 2.6. Список совместно используемых приложений системы взаимодействия

После этого пользователи любого из этих приложений могут подобрать в свое обсуждение участников из обоих объединенных приложений. При этом система взаимодействия будет различать Соколову Ольгу Владимировну из первого приложения и Соколову Наталью Павловну из второго приложения (рис. 2.7).

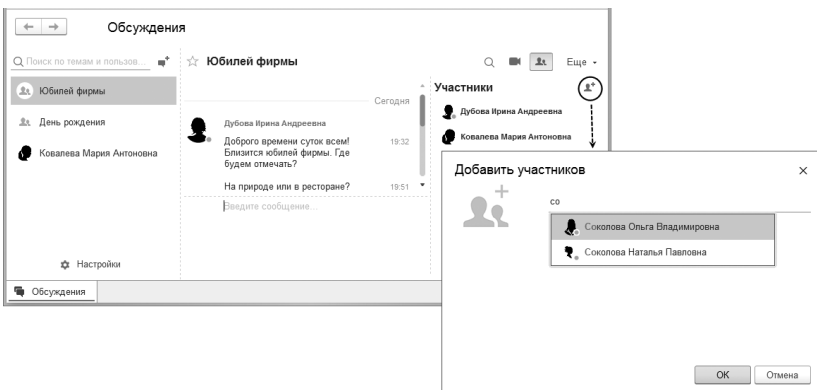


Рис. 2.7. Подбор пользователей для группового обсуждения из объединенных приложений

А при открытии основной формы системы взаимодействия Соколова Ольга Владимировна и Соколова Наталья Павловна будут видеть только свои обсуждения (рис. 2.8).



Рис. 2.8. Основная форма системы взаимодействия у пользователей объединенных приложений с одинаковыми именами

В данном случае мы видим, что у этих пользователей существуют как обсуждения внутри «своих» приложений, так и обсуждения между совместно используемыми приложениями (обсуждение «Юбилей фирмы»).

Необходимо учитывать, что при объединении нескольких приложений эти приложения начинают работать по схеме «все со всеми». То есть каждое из объединенных приложений получает

возможность взаимодействовать с любым другим приложением из списка объединенных.

Причем отмеченные для объединения приложения формируют разнообразные пары друг с другом. В нашем примере мы отметили два приложения, поэтому в списке совместного использования приложений появилась одна пара (см. рис. 2.6). Если же объединяемых приложений будет три, то образуются три пары совместно используемых приложений (Приложение1 – Приложение2, Приложение1 – Приложение3, Приложение2 – Приложение3). И так далее, по аналогии.

Теперь перейдем к предметному обсуждению между объединенными приложениями. Как уже говорилось, документы по тарификации сотрудников передаются по плану обмена из приложения «Кадры – Тарификация» (первое приложение) в приложение «Кадры – Зарплата» (второе приложение), и там на основе этих документов рассчитывается зарплата сотрудников.

Поскольку при настройке совместного использования этих приложений мы установили флажок у параметра Сопоставление предметных обсуждений (см. рис. 2.6), то один и тот же документ, переданный по плану обмена (обладающий одинаковой объектной ссылкой), можно обсуждать в разных приложениях. Посмотрим, как это выглядит, на примере.

Представим себе такую ситуацию. К бухгалтеру из приложения «Кадры – Зарплата» приходит некий сотрудник с вопросом: почему ему не начислили надбавку за ученую степень, хотя он передал в кадры все необходимые документы? Она обещает разобраться.

Для этого во втором приложении она открывает обсуждение «один на один» с расчетчиком зарплаты и задает ей соответствующий вопрос. Расчетчик отвечает, что зарплата рассчитана в соответствии с документом тарификации (рис. 2.9).

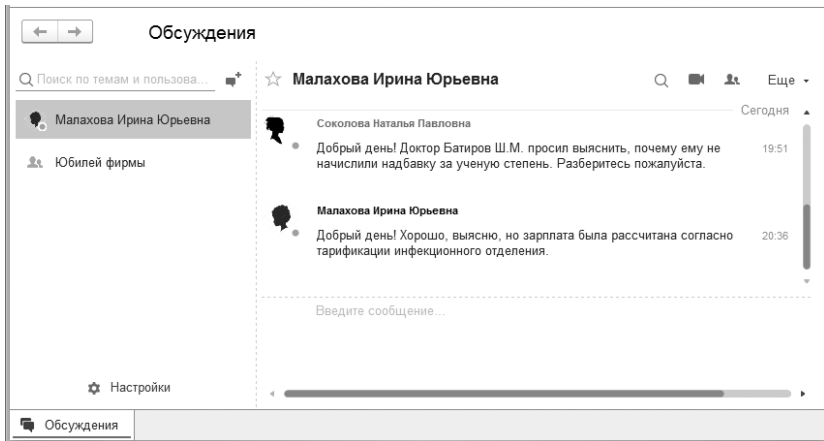


Рис. 2.9. Обсуждение «один на один» вопроса по зарплате сотрудника между бухгалтером и расчетчиком

Расчетчик открывает соответствующий тарификационный документ, убеждается, что все сделала правильно, открывает обсуждение этого документа и задает вопрос уже экономисту из приложения «Кадры – Тарификация», которая рассчитывала тарификацию сотрудника. Кроме того, она оповещает об этом бухгалтера, и та включает режим наблюдения за обсуждением документа (рис. 2.10, 2.11).

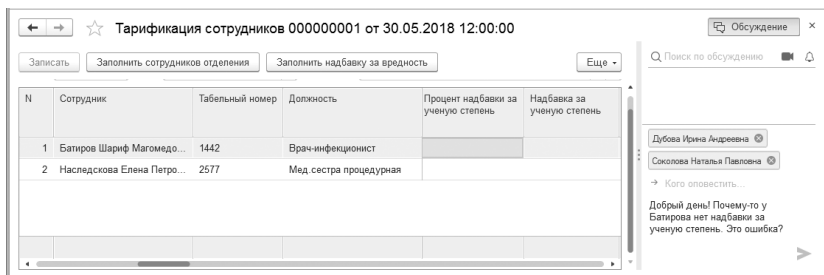


Рис. 2.10. Обсуждение вопроса по зарплате сотрудника со стороны расчетчика зарплаты

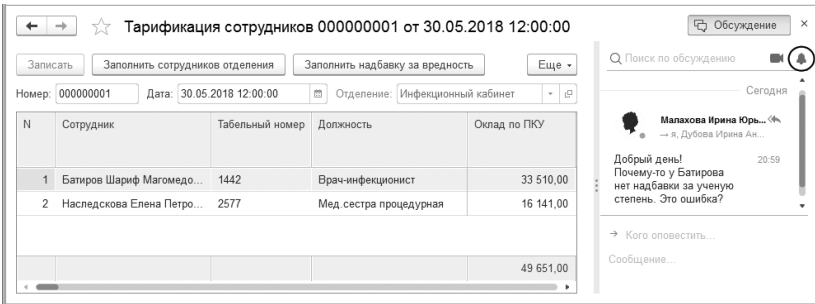


Рис. 2.11. Наблюдение за обсуждением со стороны бухгалтера

Экономист продолжает предметное обсуждение этого документа с кадровиком внутри первого приложения. Выясняется, что причина ошибки в том, что кадровик не внесла информацию об ученой степени в карточку сотрудника (рис. 2.12).

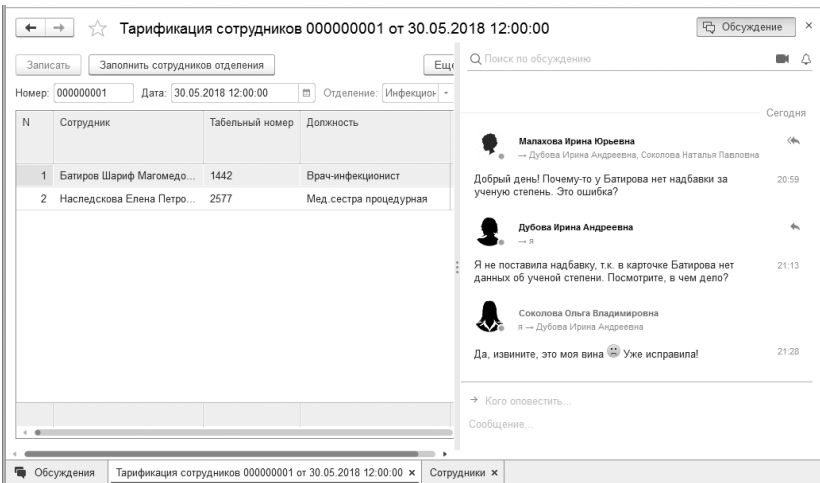


Рис. 2.12. Обсуждение вопроса по зарплате сотрудника со стороны кадровика

Поскольку у бухгалтера из второго приложения есть права на просмотр документов тарификации и у нее включено наблюдение за обсуждением этого документа, она получает уведомления о любых новых сообщениях в обсуждении и, таким образом,

может наблюдать за процессом решения спорного вопроса (рис. 2.13).

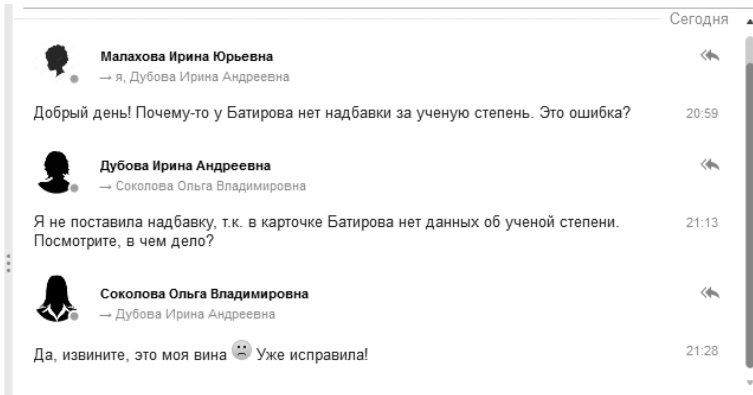


Рис. 2.13. Наблюдение за обсуждением со стороны бухгалтера

Дальше кадровик исправляет ошибку, экономист пересчитывает тарификацию, исправленный документ по плану обмена передается во второе приложение, о чем уведомляется расчетчик. Она пересчитывает зарплату сотрудника и уведомляет об этом бухгалтера. Бухгалтер видит, что проблема решена, и снимает режим наблюдения за обсуждением ошибочного документа.

Рекомендуемые сценарии подключения

В заключение хочется сказать о рекомендуемых сценариях подключения информационных баз к системе взаимодействия. В принципе, достаточно зарегистрировать приложение на сервере взаимодействия – и все заработает «само», как говорится, «из коробки».

Однако тонкость заключается в том, что пользователю не требуется отдельно регистрироваться в системе взаимодействия.

Это делает сама платформа незаметно для него путем сопоставления пользователей информационной базы и пользователей системы взаимодействия. При первом входе пользователя в приложение, зарегистрированное в системе взаимодействия, платформа проверяет, есть ли соответствующий ему пользователь системы взаимодействия. Если такого пользователя нет, платформа создает его.

Поэтому для маленьких фирм с небольшим количеством пользователей можно просто зарегистрировать информационную базу на сервере взаимодействия и больше ничего не делать. Если после этого все пользователи пришли на работу и вошли в приложение, то они сразу смогут обмениваться сообщениями.

Для больших фирм, когда сотрудников много и они приходят в разное время, кто-то болеет, кто-то в отпуске, в командировке, лучше заранее подготовиться. Написать обработку, которая для всех пользователей информационной базы создаст соответствующих пользователей системы взаимодействия. Потом зарегистрировать приложение на сервере взаимодействия и запустить эту обработку. В противном случае нельзя будет отправить сообщение тому человеку, который еще не пришел на работу или который в командировке/отпуске.

Если же предполагается совместно использовать несколько приложений и при этом сопоставлять пользователей по ключу сопоставления, то такая обработка необходима еще и для того, чтобы при массовом создании пользователей системы взаимодействия проставить им правильные ключи сопоставления. В этом случае сначала нужно подключить одну информационную базу и запустить в ней эту обработку. Потом подключить вторую информационную базу и запустить в ней обработку. И только после этого устанавливать совместное использование этих приложений. При другой последовательности действий могут появиться либо «пользователи без людей», либо несколько пользователей для одного человека.

Программа «1С:Предприятие – оповещения и запуск»

Программа «1С:Предприятие – оповещения и запуск» (или агент клиентского приложения) – это необязательный компонент системы взаимодействия. Она входит в состав дистрибутива платформы, но не устанавливается на компьютер автоматически.

Система взаимодействия будет работать и без нее, но с помощью агента клиентского приложения можно оперативно реагировать на новые сообщения (даже в том случае, если приложение не запущено), а также просто и удобно выполнять настройку оповещений.

Для установки программы «1С:Предприятие – оповещения и запуск» нужно выполнить команду Настройки из формы Обсуждения и в появившемся окне нажать на ссылку Установить (рис. 2.14).

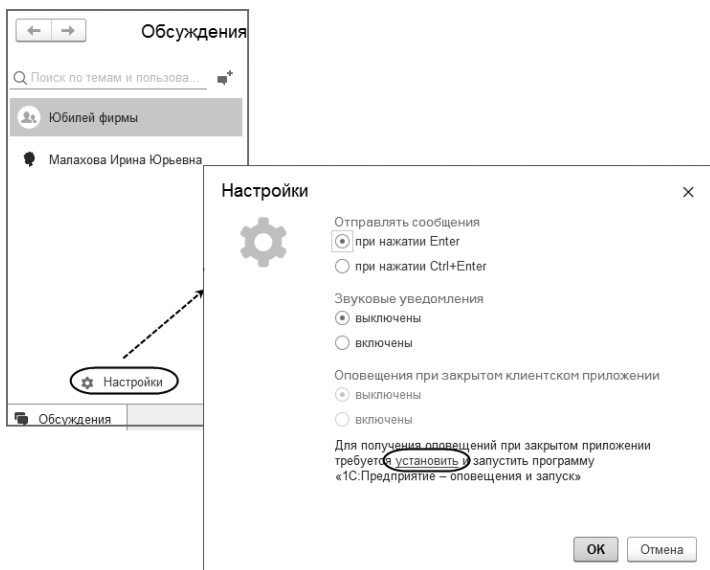


Рис. 2.14. Установка агента клиентского приложения

После установки программы у пользователя, который проводил установку, будут включены оповещения при закрытом клиентском приложении, а также появится наименование клиентского приложения в контекстном меню программы «1С:Предприятие – оповещения и запуск». Это наименование можно произвольно изменить (рис. 2.15).

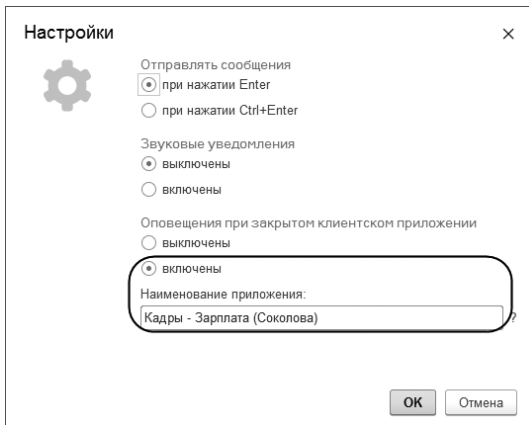


Рис. 2.15. Установка агента клиентского приложения

Для других пользователей системы взаимодействия программу устанавливать уже не требуется, достаточно только отметить опцию Включены в группе Оповещения при закрытом клиентском приложении и задать наименование приложения в программе или согласиться с предложенным по умолчанию.

После этого в области системных уведомлений появится значок программы «1С:Предприятие – оповещения и запуск». При нажатии на этот значок всплывет меню программы, содержащее список клиентских приложений, подключенных к программе, стартер «1С:Предприятия», а также возможность настройки оповещений каждого из приложений и отключения приложений от программы. Если есть проблемы при подключении к серверу взаимодействия, внешний вид значка в области

системных уведомлений и меню программы сигнализирует об этом (рис. 2.16).

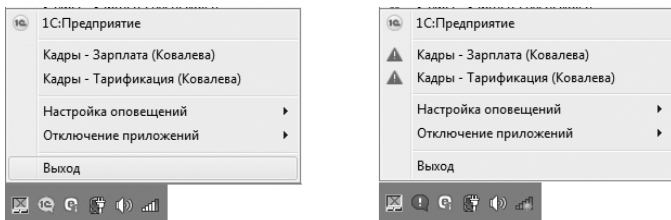


Рис. 2.16. Значок и меню агента клиентского приложения

В процессе взаимодействия пользователей, подключенных к агенту клиентского приложения, значок программы в области системных уведомлений показывает количество непросмотренных оповещений, а также это количество отражается в контекстном меню рядом с наименованием клиентского приложения (рис. 2.17).

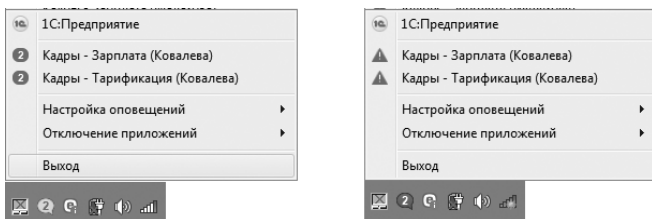


Рис. 2.17. Значок и меню агента клиентского приложения

Итак, при подключении клиентских приложений к программе «1С:Предприятие – оповещения и запуск» пользователям системы взаимодействия предоставляется несколько удобных возможностей.

Во-первых, они не пропустят оповещения, даже при закрытом клиентском приложении. Количество неп прочитанных оповещений будет показывать значок в области системных уведомлений (см. рис. 2.17). При нажатии на имя клиентского приложения

в меню программы оно запустится (если оно не было запущено), или развернется (если оно было свернуто), или активизируется (если оно было неактивно).

Во-вторых, если у пользователя запущено несколько клиентских приложений, то при одновременном получении оповещений из этих приложений они будут отображаться централизованно, не перекрывая друг друга (рис. 2.18).

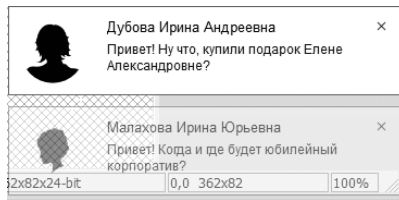


Рис. 2.18. Централизованное отображение оповещений пользователя

Помимо этого из меню программы можно отменить показ оповещений как для отдельного клиентского приложения, так и сразу для всех приложений сразу (рис. 2.19).



Рис. 2.19. Настройка показа оповещений

А также при необходимости можно прямо из меню агента клиентского приложения запустить стартер «1С:Предприятия».

ГЛАВА 3

ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ И СЦЕНАРИИ ПРИМЕНЕНИЯ СИСТЕМЫ ВЗАИМОДЕЙСТВИЯ

В этой главе мы рассмотрим конкретные примеры использования системы взаимодействия на основе реальных задач, которые возникают у пользователей, и посмотрим, как система взаимодействия помогает решать эти задачи.

За основу возьмем работу кадровика в некой организации, например в интернет-магазине. В интернет-магазин принимается много сотрудников на должности курьеров, менеджеров и т.д.

Сотрудники принимаются на испытательный срок, и когда этот срок подходит к концу, руководитель должен принять решение: брать ли сотрудника на постоянную работу или нет.

Для этого кадровик общается с руководителем интернет-магазина и выясняет у него этот вопрос. В случае положительного ответа кадровик оповещает сотрудника, что он принят, и поздравляет его. И затем представляет нового сотрудника остальным сотрудникам магазина.

«Сложность» заключается в том, что при этом должны взаимодействовать два приложения: кадровая база (куда имеют доступ только руководитель и кадровик) и база интернет-магазина (где работают все остальные сотрудники). Испытательный срок и все остальные кадровые данные сотрудников хранятся в кадровой базе. Поэтому обсуждение кадровика с руководителем происходит внутри этой базы, а вот для общения кадровика с сотрудниками интернет-магазина эти две базы нужно объединить в рамках системы взаимодействия (речь идет об их совместном использовании).

Реализуем этот сценарий работы сначала интерактивно, а затем выполним то же самое программным образом.

Интерактивно

В этом варианте все делается вручную, но зато нам не нужно вообще ничего программировать!

ПРИМЕЧАНИЕ

Примеры интерактивного взаимодействия кадровика с руководителем и кадровика с сотрудниками интернет-магазина можно посмотреть в конфигурациях «ИнтерактивноеВзаимодействиеКадры» и «ИнтерактивноеВзаимодействиеИнтернетМагазин», которые прилагаются к книге на компакт-диске.

Прежде всего нужно зарегистрировать на сервере взаимодействия наши информационные базы, как показывалось в разделе «Регистрация приложений на сервере взаимодействия» на стр. 36, и настроить их совместное использование (рис. 3.1).

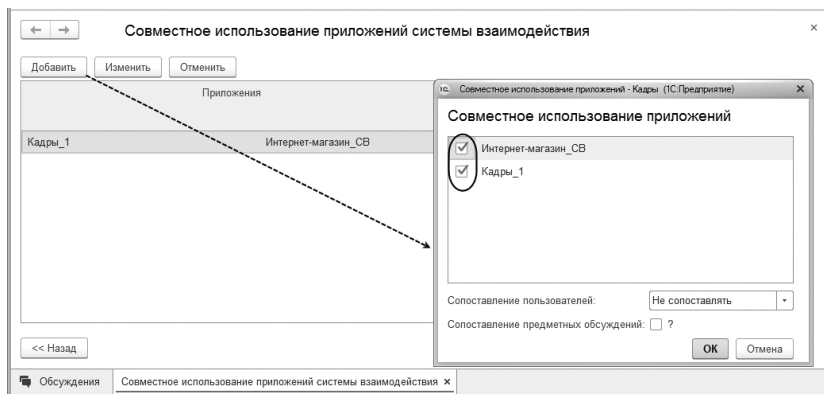


Рис. 3.1. Совместное использование приложений

При объединении приложений пользователи и предметные обсуждения не сопоставляются, потому что нам это не нужно, так как в данном примере у нас нет одинаковых пользователей и предметов обсуждения в разных базах.

Предположим, каждый день кадровик вручную проверяет информацию – у какого сотрудника заканчивается испытательный срок. Затем он создает предметное обсуждение в карточке этих сотрудников с руководителем и задает ему вопрос о приеме сотрудника на постоянную работу (рис. 3.2).

Получив оповещение о новом сообщении от кадровика, руководитель нажимает на него и сразу же попадает в предметное обсуждение сотрудника. Изучив кадровые данные в карточке сотрудника, различные отчеты о работе сотрудника и т. п., руководитель посылает свое решение кадровику (рис. 3.3).

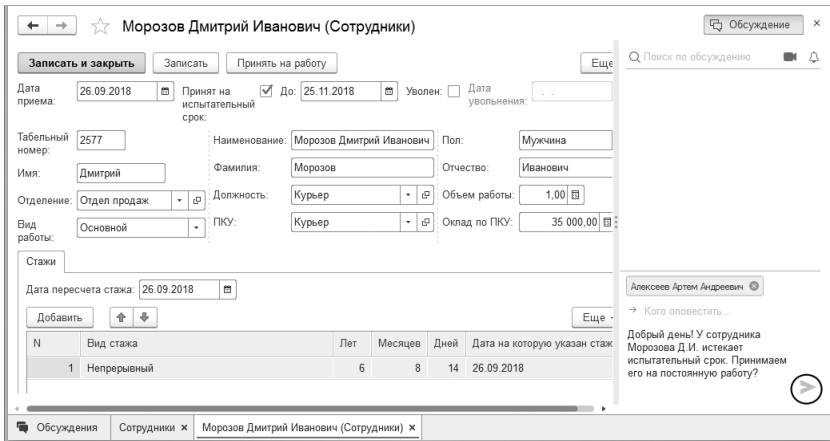


Рис. 3.2. Предметное обсуждение кадровика с руководителем в карточке сотрудника

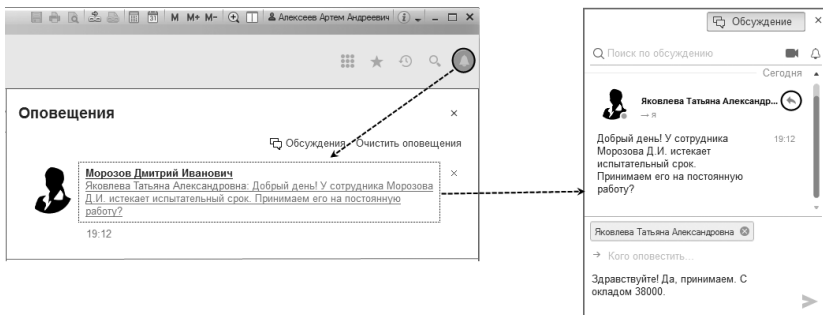


Рис. 3.3. Решение руководителя в предметном обсуждении

В зависимости от ответа руководителя кадровик вносит соответствующие изменения в карточку сотрудника, готовит приказ о его приеме на работу, меняет оклад и т. п. (рис. 3.4).

После этого кадровик создает обсуждения «один на один» с каждым из сотрудников, по которым получено положительное решение, и поздравляет их с принятием на постоянную работу, а также вкладывает в сообщение поздравительную картинку. Кроме того, сообщение содержит гиперссылку (URL в виде текстовой

строки) на информацию для новых сотрудников, которая размещена на внутреннем сайте организации (рис. 3.5).

Морозов Дмитрий Иванович (Сотрудники) *

Записать и закрыть | Записать | Принять на работу | Еще -

Дата приема: 26.09.2018 | Принять на испытательный срок: Да: ... | Уволен: | Дата увольнения: ...

Табельный номер: 2577 | Наименование: Морозов Дмитрий Иванович | Пол: Мужчнина

Имя: Дмитрий | Фамилия: Морозов | Отчество: Иванович

Отделение: Отдел продаж | Должность: Курьер | Объем работы: 1,00

Вид работы: Основной | ПКУ: Курьер | Заработок по ПКУ: 38 000,00

Стажи

Дата пересчета стажа: 26.09.2018

N	Вид стажа	Лет	Месяцев	Дней	Дата на которую указан стаж
1	Непрерывный	6	8	14	26.09.2018

Обсуждения | Сотрудники | Морозов Дмитрий Иванович (Сотрудники) * x

Рис. 3.4. Изменения в карточке сотрудника

Кадры (ИС/Предприятие) | Яковлева Татьяна Александровна

Главное | Кадры

Сотрудники | Штатное расписание | Отделения | Должности | Профессионально-квалификационные уровни | Виды стажа

Обсуждения

Морозов Дмитрий Иванович

Яковлева Татьяна Александровна | 21.11.2018

Поздравляем! С 26.11.2018 Вы приняты в постоянный штат сотрудников отдела продаж на должность курьера. Добро пожаловать в наш коллектив 😊. Дополнительную информацию смотрите на сайте <http://infosever/newbie.htm>

Введите сообщение.

Интернет-магазин (ИС/Предприятие) | Морозов Дмитрий Иванович

Обсуждения | Заказы | Обслуживание заказов | Клиенты | Товары | Цвета | Размеры | Еще - | Создать - | Отчеты - | Сервис -

Обсуждения

Яковлева Татьяна Александровна

Яковлева Татьяна Александровна | 21.11.2018

Поздравляем! С 26.11.2018 Вы приняты в постоянный штат сотрудников отдела продаж на должность курьера. Добро пожаловать в наш коллектив 😊. Дополнительную информацию смотрите на сайте <http://infosever/newbie.htm>

Рис. 3.5. Обсуждение с сотрудником «один на один» с поздравлением о принятии на работу

Хотя сообщение посылается в другую базу, подбор сотрудника делается так же, как обычно (по первым буквам наименования в строке поиска обсуждений), так как объединенные приложения обладают единым списком пользователей

Затем кадровик создает групповое обсуждение («Новый сотрудник») со всеми остальными сотрудниками из обеих баз, в котором представляет нового сотрудника остальным (рис. 3.6).

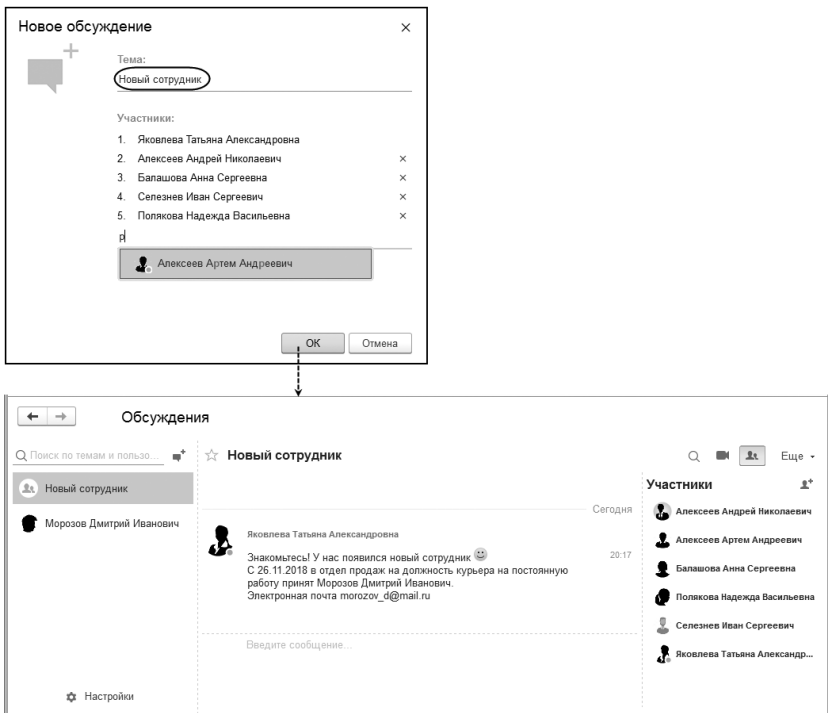


Рис. 3.6. Групповое обсуждение «Новый сотрудник»

В результате все пользователи, указанные в списке участников группового обсуждения на рис. 3.6, узнают о новом сотруднике (рис. 3.7).

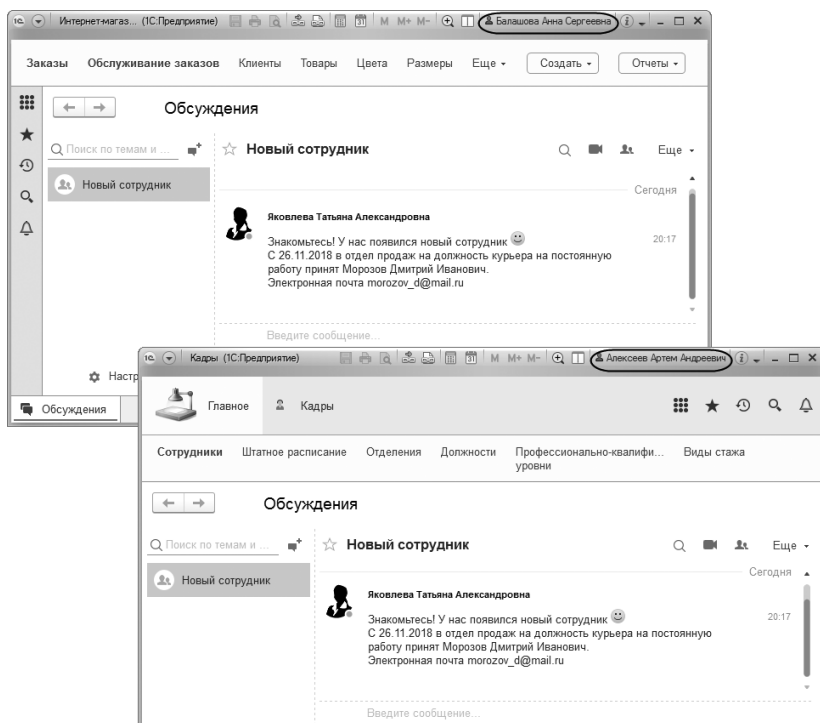


Рис. 3.7. Групповое обсуждение «Новый сотрудник»

Программно

Вся функциональность системы взаимодействия доступна не только интерактивно, но и программно с помощью менеджера системы взаимодействия, который представлен свойством глобального контекста СистемаВзаимодействия. То есть через точку от менеджера мы получаем доступ к различным объектам системы взаимодействия и можем использовать их методы и свойства.

В программном варианте работы того же сценария мы «упростим жизнь» кадровику. Пусть кадровик каждое утро получает список сотрудников, у которых заканчивается испытательный срок, уже не вручную, а с помощью регламентного задания.

ПРИМЕЧАНИЕ

Примеры реализации программного взаимодействия кадровика с руководителем и кадровика с сотрудниками интернет-магазина можно посмотреть в конфигурациях «ПрограммноеВзаимодействиеКадры» и «ПрограммноеВзаимодействиеИнтернетМагазин», которые прилагаются к книге на компакт-диске.

Отправка напоминаний-сообщений в групповое обсуждение

Итак, создадим предопределенное регламентное задание `СотрудникиНаПрием`, зададим ему расписание (например, выполнять каждый день, с 10:00:00 один раз в день) и укажем имя метода, который будет выполняться по этому расписанию.

Процедуру `ПолучитьСписокСотрудниковНаПрием()`, выполняющуюся с помощью регламентного задания, поместим в общий неглобальный серверный модуль `РаботаССотрудниками` (с включенным флажком `Вызов сервера`) и заполним следующим образом (листинг 3.1).

Листинг 3.1. Процедура «ПолучитьСписокСотрудниковНаПрием»

Процедура `ПолучитьСписокСотрудниковНаПрием()` Экспорт

```
Запрос = Новый Запрос;  
Запрос.Текст =  
    "ВЫБРАТЬ  
    |     Сотрудники.Ссылка КАК Ссылка,  
    |     Сотрудники.Наименование КАК Наименование,  
    |     Сотрудники.ДатаПриема КАК ДатаПриема,  
    |     Сотрудники.Отделение КАК Отделение,  
    |     Сотрудники.Должность КАК Должность,
```

```

| Сотрудники.ДатаОкончанияИспСрока КАК ДатаОкончанияИспСрока
| ИЗ
| Справочник.Сотрудники КАК Сотрудники
| ГДЕ
| Сотрудники.Уволен = ЛОЖЬ
| И Сотрудники.ПометкаУдаления = ЛОЖЬ
| И Сотрудники.ИспытательныйСрок = ИСТИНА
| И Сотрудники.ДатаОкончанияИспСрока < &ДатаИспСрока
|
| УПОРЯДОЧИТЬ ПО
| ДатаОкончанияИспСрока";

```

Запрос.УстановитьПараметр("ДатаИспСрока", ТекущаяДата() + 5 * 24 * 60 * 60);

РезультатЗапроса = Запрос.Выполнить();

Если РезультатЗапроса.Пустой() Тогда

Возврат;

КонецЕсли;

Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда

Возврат;

КонецЕсли;

ИдентификаторПользователяСВКадровик =

СистемаВзаимодействия.ИдентификаторТекущегоПользователя();

// Напоминание кадровику о сотрудниках с истекающим испытательным сроком

КлючОбсуждения = "ИспытательныйСрок";

Обсуждение = СистемаВзаимодействия.ПолучитьОбсуждение(КлючОбсуждения);

Если Обсуждение = Неопределено Тогда

Обсуждение = СистемаВзаимодействия.СоздатьОбсуждение();

Обсуждение.Ключ = КлючОбсуждения;

Обсуждение.Заголовок = "Сотрудники на прием";

Обсуждение.Участники.Добавить(ИдентификаторПользователяСВКадровик);

Обсуждение.Записать();

КонецЕсли;

Сообщение = СистемаВзаимодействия.СоздатьСообщение(Обсуждение.Идентификатор);

Текст = "Список сотрудников с истекающим испытательным сроком" + Символы.ПС;

СписокСотрудников = Новый Массив;

Выборка = РезультатЗапроса.Выбрать();

Пока Выборка.Следующий() Цикл

СписокСотрудников.Добавить(Выборка.Ссылка);

Текст = Текст + Символы.ПС;

Текст = Текст + "Сотрудник: " + ПолучитьНавигационнуюСсылку(Выборка.Ссылка) + Символы.ПС +

"Отделение: " + ПолучитьНавигационнуюСсылку(Выборка.Отделение) +

"Должность: " + ПолучитьНавигационнуюСсылку(Выборка.Должность) + Символы.ПС +

"Дата приема: " + Формат(Выборка.ДатаПриема, "ДФФ=Д") + Символы.ПС +

```
        "Дата окончания испытательного срока:" +  
        Формат(Выборка.ДатаОкончанияИспСрока, "ДФФ=Д");  
    Текст = Текст + Символы.ПС;  
    КонецЦикла;  
  
    Сообщение.Текст = Новый ФорматированнаяСтрока(Текст);  
    // заполняем данные сообщения списком ссылок сотрудников на прием  
    Сообщение.Данные = СписокСотрудников;  
    // добавляем действия сообщения для кадровика  
    Сообщение.Действия.Добавить("Request", "Запросить результаты у руководителя");  
    Сообщение.Записать();
```

КонецПроцедуры

Поясним текст процедуры.

Сначала мы запросом получаем список сотрудников, находящихся на испытательном сроке, который заканчивается через пять дней. Если таких сотрудников на данный момент нет, прекращаем работу процедуры.

Затем с помощью метода `ИнформационнаяБазаЗарегистрирована()` менеджера системы взаимодействия (в дальнейшем уточнение с именем менеджера будет опускаться) определяем, зарегистрирована ли информационная база в системе взаимодействия. Эта проверка необходима, так как если база не зарегистрирована или регистрация была отменена, то в дальнейших действиях нет смысла и они завершатся ошибкой.

Далее с помощью метода `ИдентификаторТекущегоПользователя()` получаем идентификатор текущего пользователя системы взаимодействия, то есть кадровика. Именно от его имени запускается регламентное задание, и именно ему с помощью системы взаимодействия должно прийти сообщение со списком сотрудников, у которых истекает испытательный срок.

Любое сообщение системы взаимодействия (далее просто «сообщение») существует не само по себе, а в рамках какого-то обсуждения системы взаимодействия (далее просто «обсуждение»). Для получения списка сотрудников с истекающим

испытательным сроком (на кадровом сленге они называются «сотрудники на прием») мы будем использовать обсуждение с заголовком Сотрудники на прием с ключом ИспытательныйСрок.

Свойство Ключ объекта ОбсуждениеСистемыВзаимодействия содержит уникальный строковый ключ обсуждения, который устанавливается при первой записи обсуждения. Уникальность ключа должна соблюдаться в пределах информационной базы в разрезе всех независимых разделителей.

С помощью метода ПолучитьОбсуждение() мы проверяем, существует ли обсуждение с ключом «ИспытательныйСрок». Если нет (метод вернул Неопределено), то мы создаем новое обсуждение методом СоздатьОбсуждение().

В результате создается объект ОбсуждениеСистемыВзаимодействия. У нового обсуждения мы устанавливаем свойства Ключ и Заголовок.

Затем добавляем в список участников обсуждения нашего кадровика. Для этого мы получаем коллекцию идентификаторов пользователей системы взаимодействия (Обсуждение.Участники) и добавляем туда идентификатор кадровика, который мы получили ранее, как идентификатор текущего пользователя системы взаимодействия.

И в заключение записываем новое обсуждение методом Записать(). После записи обсуждения становится доступен его идентификатор.

В случае если обсуждение с ключом ИспытательныйСрок уже существует, то метод ПолучитьОбсуждение() вернет это обсуждение и мы также будем знать его идентификатор (Обсуждение.Идентификатор), который нам понадобится для того, чтобы однозначно прикрепить сообщение со списком сотрудников на прием к обсуждению.

Итак, с помощью метода `СоздатьСообщение()` мы создаем новое сообщение (объект `СообщениеСистемыВзаимодействия`) в обсуждении, идентификатор которого нам известен.

Теперь нам нужно заполнить текст сообщения результатами выполнения запроса, описанного в начале процедуры. Обходя выборку из результата запроса в цикле, мы заполняем текст, который будет затем помещен в сообщение, данными выполнения запроса. Ссылки на сотрудников, отделения и должности мы представляем в качестве навигационных ссылок, которые в тексте сообщения будут служить гиперссылками для перехода к конкретным данным.

Кроме того, в цикле обхода выборки мы заполняем массив `СписокСотрудников` ссылками сотрудников на прием. Этот массив после выхода из цикла мы сохраняем в свойстве `Данные` нашего сообщения. Таким образом, помимо текста сообщение будет содержать еще и данные, которые будут использоваться при обработке действий сообщения.

В свойстве `Текст` мы сохраняем в виде форматированной строки текст, заполненный данными запроса.

После этого добавляем в список действий сообщения (`Сообщение. Действия`) элемент с представлением «Запросить результаты у руководителя» и значением "Request". Представления действий будут отображаться в виде гиперссылок при отображении сообщения, а сами значения будут передаваться в обработчик действий сообщения, который мы рассмотрим ниже (см. листинг 3.4).

И в заключение записываем сообщение методом `Записать()`. Сразу после этого оно будет отправлено кадровику.

В нашем случае мы создали по умолчанию групповое обсуждение с одним-единственным участником.

Таким образом, в результате выполнения регламентного задания в первый раз в обсуждениях кадровика появится обсуждение

Сотрудники на прием с сообщением, содержащим данные сотрудников с истекающим испытательным сроком, и гиперссылкой Запросить результаты у руководителя (рис. 3.8).

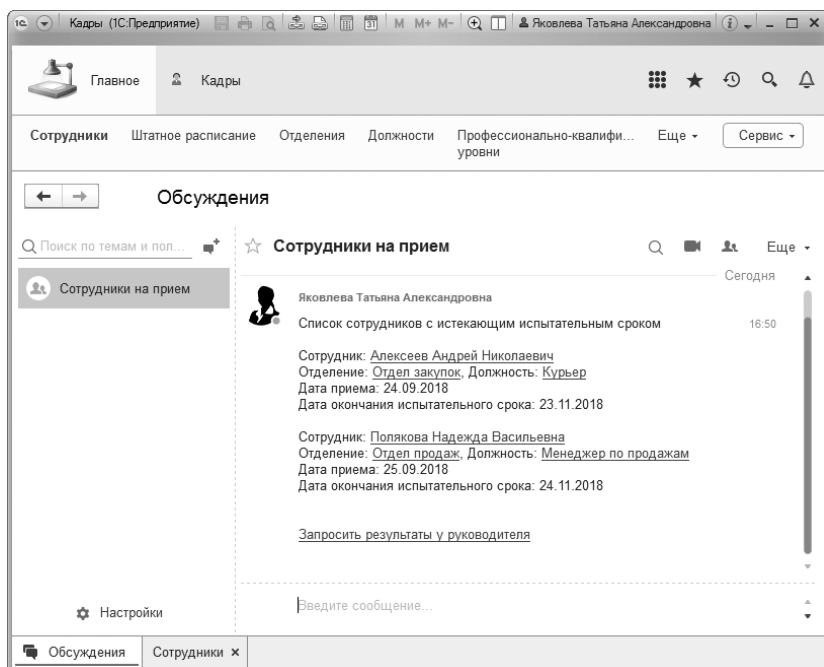


Рис. 3.8. Групповое обсуждение «Сотрудники на прием»

При регулярных выполнениях регламентного задания, в зависимости от наличия списка сотрудников на прием, в уже существующее обсуждение Сотрудники на прием будут добавляться все новые сообщения.

Обработка действий сообщения

При нажатии на гиперссылку Запросить результаты у руководителя будут организованы предметные обсуждения кадровика с руководителем в карточках этих сотрудников.

Чтобы это реализовать, мы должны при начале работы системы подключить процедуру-обработчик, вызываемую при нажатии на гиперссылку в сообщении, заданную в свойстве сообщения Действия. Для этого нам нужно описать эту процедуру в клиентском модуле с помощью описания оповещения и подключить ее методом ПодключитьОбработчикДействияСообщения() менеджера системы взаимодействия. А затем вызвать процедуру этого модуля из модуля приложения при начале работы системы.

Итак, создадим общий неглобальный клиентский модуль РаботаССотрудникамиКлиент и поместим в нем экспортную процедуру ПриНачалеРаботыСистемы() (листинг 3.2), которую затем будем вызывать из модуля приложения (см. листинг 3.3).

Листинг 3.2. Процедура «ПриНачалеРаботыСистемы» общего модуля «РаботаССотрудникамиКлиент»

```
Процедура ПриНачалеРаботыСистемы() Экспорт
    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат;
    КонецЕсли;

    Обработчик = Новый ОписаниеОповещения("ОбработкаДействияСообщения", ЭтотОбъект);
    СистемаВзаимодействия. ПодключитьОбработчикДействияСообщения(Обработчик);

КонецПроцедуры
```

В этой процедуре мы описываем обработчик оповещения и подключаем процедуру ОбработкаДействияСообщения(), которая будет вызываться при нажатии гиперссылки (выполнения действия) в сообщении.

Затем подключаем этот обработчик при начале работы приложения (листинг 3.3).

Листинг 3.3. Процедура «ПриНачалеРаботыСистемы» модуля приложения

```
Процедура ПриНачалеРаботыСистемы()
```

```
    РаботаССотрудникамиКлиент.ПриНачалеРаботыСистемы();
```

```
КонецПроцедуры
```

Таким образом, при выполнении любого действия в любом сообщении системы взаимодействия будет выполняться процедура `ОбработкаДействияСообщения()`.

Первым параметром в обработчик оповещения передается копия объекта `СообщениеСистемыВзаимодействия`, то есть того сообщения, в котором нажата гиперссылка действия. Вторым параметром передается `Действие` – произвольное значение, заданное для нажатой гиперссылки.

Экспортную процедуру `ОбработкаДействияСообщения()` также поместим в модуле `РаботаССотрудникамиКлиент` и заполним ее следующим образом (листинг 3.4).

Листинг 3.4. Процедура «ОбработкаДействияСообщения»

```
Процедура ОбработкаДействияСообщения(Сообщение, Действие, ДопПараметры) Экспорт
```

```
    Если Действие = "Request" Тогда
```

```
        РаботаССотрудниками.СогласоватьПринятиеСРуководителем(Сообщение.Данные);
```

```
    КонецЕсли;
```

```
КонецПроцедуры
```

В этом обработчике, в случае если параметр `Действие` имеет значение "Request", мы вызываем процедуру `СогласоватьПринятиеСРуководителем()`, в которой и будет организовано предметное обсуждение кадровика с руководителем в карточке сотрудников по поводу их приема на постоянную работу.

Предметное обсуждение сотрудников

В параметре Сообщение. Данные мы передаем в процедуру массив ссылок на сотрудников с истекающим испытательным сроком, который мы заполнили при создании сообщения (см. листинг 3.1). Саму процедуру СогласоватьПринятиеСРуководителем() поместим в модуле РаботаССотрудниками и заполним следующим образом (листинг 3.5).

Листинг 3.5. Процедура «СогласоватьПринятиеСРуководителем»

```
Процедура СогласоватьПринятиеСРуководителем(СписокСотрудников) Экспорт
```

```
Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда  
    Возврат;  
КонецЕсли;
```

```
УстановитьПривилегированныйРежим(Истина);
```

```
Для Каждого ПользовательИБ ИЗ ПользователиИнформационнойБазы.ПолучитьПользователей() Цикл
```

```
    Если ПользовательИБ.Роли.Содержит(Метаданные.Роли.Кадровик) Тогда
```

```
        ИдентификаторПользователяСВКадровик =  
            СистемаВзаимодействия.ПолучитьИдентификаторПользователя(  
                ПользовательИБ.УникальныйИдентификатор);
```

```
        КонецЕсли;
```

```
        Если ПользовательИБ.Роли.Содержит(Метаданные.Роли.Руководитель) Тогда
```

```
            ИдентификаторПользователяСВРуководитель =  
                СистемаВзаимодействия.ПолучитьИдентификаторПользователя(  
                    ПользовательИБ.УникальныйИдентификатор);
```

```
        КонецЕсли
```

```
    КонецЦикла;
```

```
Для Каждого СотрудникНаПрием ИЗ СписокСотрудников Цикл
```

```
    // Обсуждение для сотрудника
```

```
    НавигационнаяСсылка = ПолучитьНавигационнуюСсылку(СотрудникНаПрием);
```

```
    Контекст = Новый КонтекстОбсужденияСистемыВзаимодействия(НавигационнаяСсылка);
```

```
    ОтборСВ = Новый ОтборОбсужденийСистемыВзаимодействия();
```

```
    ОтборСВ.КонтекстноеОбсуждение = Истина;
```

```
    ОтборСВ.КонтекстОбсуждения = Контекст;
```

```
    Обсуждения = СистемаВзаимодействия.ПолучитьОбсуждения(ОтборСВ);
```

```
    Если Обсуждения.Количество() = 0 Тогда
```

```
        // Если не найдено, создаем новое
```

```

Обсуждение = СистемаВзаимодействия.СоздатьОбсуждение();
Обсуждение.КонтекстОбсуждения = Контекст;
Обсуждение.Ключ = НавигационнаяСсылка;
Обсуждение.Записать();
ИдентификаторОбсуждения = Обсуждение.Идентификатор;
Иначе
    ИдентификаторОбсуждения = Обсуждения[0].Идентификатор;
КонецЕсли;

Сообщение = СистемаВзаимодействия.СоздатьСообщение(ИдентификаторОбсуждения);
Сообщение.Автор = ИдентификаторПользователяСВКадровик;
Сообщение.Получатели.Добавить(ИдентификаторПользователяСВРуководитель);
Сообщение.Текст = "Добрый день! У сотрудника: " + СотрудникНаПрием.Наименование +
    " заканчивается испытательный срок. Принимаем его на постоянную работу?";

// добавляем действия сообщения для руководителя
Сообщение.Действия.Добавить("Ассерп", "Да");
Сообщение.Действия.Добавить("Cancel", "Нет");

// заполняем данные сообщения
СтруктураДанных = Новый Структура;
СтруктураДанных.Вставить("Обсуждение", ИдентификаторОбсуждения);
СтруктураДанных.Вставить("Отправитель", ИдентификаторПользователяСВКадровик);
СтруктураДанных.Вставить("Адресат", ИдентификаторПользователяСВРуководитель);
Сообщение.Данные = СтруктураДанных;

Сообщение.Записать();
КонецЦикла;
КонецПроцедуры

```

Поясним текст процедуры.

Сначала мы обходим в цикле коллекцию пользователей информационной базы и для пользователей с ролями Кадровик и Руководитель получаем идентификатор этих пользователей в системе взаимодействия по их уникальному идентификатору в информационной базе.

После этого мы обходим в цикле массив ссылок сотрудников на прием, переданный в процедуру в параметре СписокСотрудников, и для каждого сотрудника организуем предметное обсуждение, в котором контекст получен из навигационной ссылки на сотрудника.

В отличие от группового обсуждения, а также обсуждения «один на один», для поиска предметного обсуждения мы используем не ключ, а отбор (объект `ОтборОбсужденийСистемыВзаимодействия`). У этого отбора мы устанавливаем в истину свойство `КонтекстноеОбсуждение`, а свойство `КонтекстОбсуждения` задаем как значение объекта `КонтекстОбсужденияСистемыВзаимодействия`, полученного из навигационной ссылки на сотрудника.

Затем методом `ПолучитьОбсуждения()` мы получаем массив обсуждений с установленным отбором. В случае если задан контекст обсуждения, мы получим либо одно-единственное существующее предметное обсуждение, либо пустой массив обсуждений с указанным контекстом.

Если массив полученных обсуждений пуст, то мы создаем новое обсуждение, устанавливаем у него свойства `Ключ`, `КонтекстОбсуждения` и записываем обсуждение. После этого нам становится доступен его идентификатор.

Если же предметное обсуждение для сотрудника уже существует, мы получаем его идентификатор из единственного элемента массива найденных обсуждений – `Обсуждения[0].Идентификатор`.

После этого создаем сообщение для обсуждения с полученным идентификатором. В свойстве `Автор` запоминаем идентификатор кадровика в системе взаимодействия, а в список получателей (`Сообщение.Получатели`) добавляем идентификатор руководителя. В текст сообщения помещаем собственно вопрос по поводу приема сотрудника на постоянную работу, который кадровик задает руководителю.

Для руководителя тоже предусмотрим небольшую автоматизацию. Чтобы он не писал ответ вручную, добавим в сообщение два действия с представлениями «Да» (значение "Ассер") и «Нет» (значение "Cancel").

В свойство `Данные` поместим структуру `СтруктураДанных` с полями `Обсуждение`, `Отправитель` и `Адресат`, заполненную соответственно идентификатором обсуждения, идентификатором кадровика и идентификатором руководителя в системе взаимодействия. В заключение записываем сообщение.

Таким образом, каждый день после выполнения регламентного задания в обсуждение кадровика `Сотрудники` на прием будут добавляться сообщения со списком сотрудников с истекающим испытательным сроком и гиперссылкой `Запросить результаты у руководителя` (см. рис. 3.8). При нажатии этой гиперссылки начнет выполняться обработчик оповещения `ОбработкаДействияСообщения()`, который запустит процедуру `СогласоватьПринятиеСРуководителем()`.

В результате будут созданы предметные обсуждения всех сотрудников на прием, и руководитель получит следующие оповещения о новых сообщениях от кадровика (рис. 3.9).



Рис. 3.9. Оповещения для руководителя о новых сообщениях от кадровика

Нажав на оповещение, руководитель сразу же попадет в карточку сотрудника. В этом предметном обсуждении, сразу под текстом сообщения от кадровика, нажатием на гиперссылки «Да» или «Нет» он может выбрать соответствующие варианты ответа. В результате ответное сообщение с решением руководителя должно сразу же отправиться кадровику (рис. 3.10).

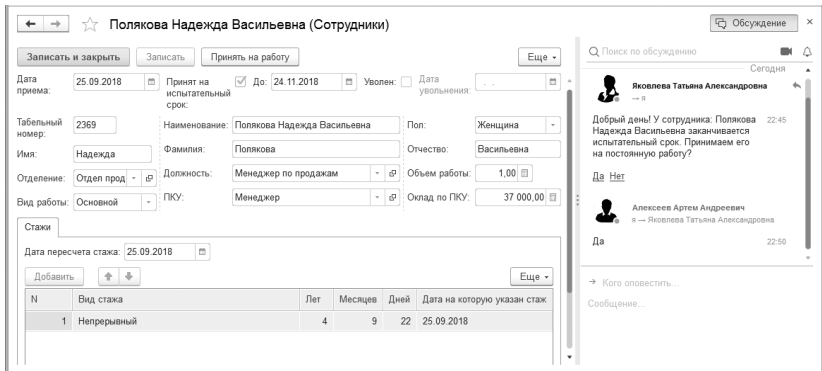


Рис. 3.10. Предметное обсуждение кадровика с руководителем в карточке сотрудника

Чтобы реализовать этот сценарий, нам нужно дополнить обработчик оповещения `ОбработкаДействияСообщения()` в модуле `РаботаССотрудникамиКлиент` следующим образом (листинг 3.6).

Листинг 3.6. Процедура «ОбработкаДействияСообщения»

Процедура `ОбработкаДействияСообщения(Сообщение, Действие, ДопПараметры)` Экспорт

```
Если Действие = "Request" Тогда
    РаботаССотрудниками.СогласоватьПринятиеСРуководителем(Сообщение.Данные);
ИначеЕсли Действие = "Accept" Тогда
    РаботаССотрудниками.РешениеРуководителя("Да", Сообщение.Данные);
ИначеЕсли Действие = "Cancel" Тогда
    РаботаССотрудниками.РешениеРуководителя("Нет", Сообщение.Данные);
КонецЕсли;
```

КонецПроцедуры

В выделенном фрагменте обработчика, в случае если параметр Действие имеет значение "Ассерт" или "Cancel", мы вызываем процедуру РешениеРуководителя() и передаем в нее собственно ответ руководителя, в параметре Сообщение. Данные мы передаем структуру данных, которую мы заполнили при создании сообщения (см. листинг 3.5).

Саму процедуру РешениеРуководителя() поместим в модуле РаботаССотрудниками и заполним следующим образом (листинг 3.7).

Листинг 3.7. Процедура «РешениеРуководителя»

Процедура РешениеРуководителя(ТекстОтвета, ДанныеСообщения) Экспорт

```
Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
    Возврат;
КонецЕсли;
```

```
Если ДанныеСообщения.Отправитель =
    СистемаВзаимодействия.ИдентификаторТекущегоПользователя() Тогда
    Возврат;
КонецЕсли;
```

```
УстановитьПривилегированныйРежим(Истина);
```

```
Сообщение = СистемаВзаимодействия.СоздатьСообщение(ДанныеСообщения.Обсуждение);
Сообщение.Автор = ДанныеСообщения.Адресат;
Сообщение.Получатели.Добавить(ДанныеСообщения.Отправитель);
Сообщение.Текст = ТекстОтвета;
Сообщение.Записать();
```

КонецПроцедуры

Поясним текст процедуры.

Сначала мы прекращаем дальнейшее выполнение процедуры для отправителя исходного сообщения с действиями «Да»/«Нет», то есть для кадровика. Идентификатор отправителя мы запомнили в структуре данных при создании исходного сообщения, которую мы передали в процедуру во втором параметре ДанныеСообщения.

В результате, если текущий пользователь системы взаимодействия является отправителем исходного сообщения (т.е. кадровиком), то при нажатии гиперссылок «Да»/«Нет» ничего не произойдет. А руководитель сможет с их помощью отправить свое решение в предметное обсуждение.

Для этого мы создаем новое сообщение в предметном обсуждении с тем же идентификатором, что и исходное сообщение (ДанныеСообщения.Обсуждение). Меняем отправителя и получателя местами – то есть автором сообщения теперь является руководитель (ДанныеСообщения.Адресат), а получателем – кадровик (ДанныеСообщения.Автор), и записываем сообщение. После записи сообщение сразу же отправляется в обсуждение.

Создание пользователей информационной базы и пользователей системы взаимодействия для принятых сотрудников

Наш пример – демонстрационный, поэтому, для того чтобы показать, как программно реализуется взаимодействие кадровика с сотрудниками интернет-магазина, мы используем ряд упрощений. А именно – предоставим кадровику возможность входа в информационную базу интернет-магазина и предположим, что, получив решение руководителя, кадровик добавляет в справочник Пользователи тех сотрудников, которых решено принять на постоянную работу. При этом он вводит сокращенное и полное имя сотрудника и выбирает из значений перечисления его должность и подразделение, в котором он будет работать (рис. 3.11).

При записи нового пользователя должен программно создаваться соответствующий ему пользователь информационной базы и пользователь системы взаимодействия. Для этого поместим в модуль справочника Пользователи обработчик события ПриЗаписи и заполним его следующим образом (листинг 3.8).

Рис. 3.11. Добавление нового сотрудника в справочник «Пользователи»

Листинг 3.8. Обработчик события «ПриЗаписи» в модуле объекта

Процедура ПриЗаписи(Отказ)

УстановитьПривилегированныйРежим(Истина);

// Создаем пользователя информационной базы для нового сотрудника
 ПользовательИнформационнойБазы = ПользователиИнформационнойБазы
 .НайтиПоИмени(СокрЛП(Код));

Если ПользовательИнформационнойБазы = Неопределено Тогда

 ПользовательИнформационнойБазы = ПользователиИнформационнойБазы
 .СоздатьПользователя();

 ПользовательИнформационнойБазы.Имя = СокрЛП(Код);

 ПользовательИнформационнойБазы.ПолноеИмя = СокрЛП(Наименование);

 Для Каждого Роль Из Метаданные.Роли Цикл

 Если Роль.Имя = Строка(Должность) Тогда

 РольПользователяИБ = Роль;

 КонецЕсли;

 КонецЦикла;

 ПользовательИнформационнойБазы.Роли.Добавить(РольПользователяИБ);

 ПользовательИнформационнойБазы.Записать();

Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
 Возврат;

КонецЕсли;

ПользовательСистемыВзаимодействия =

 СистемаВзаимодействия.СоздатьПользователя(ПользовательИнформационнойБазы);

 ПользовательСистемыВзаимодействия.Имя = СокрЛП(Код);

 ПользовательСистемыВзаимодействия.ПолноеИмя = СокрЛП(Наименование);

 ПользовательСистемыВзаимодействия.Записать();

КонецЕсли;

КонецПроцедуры

В этом обработчике мы сначала ищем пользователя информационной базы по имени (которое хранится в реквизите объекта Код).

И если пользователя информационной базы с таким именем еще не существует, то мы добавляем его. Заполняем свойства Имя, ПолноеИмя значениями реквизитов Код и Наименование нового элемента справочника Пользователи.

Затем обходим коллекцию ролей метаданных в цикле, находим роль, соответствующую строковому представлению должности нового сотрудника, и добавляем ее новому пользователю информационной базы.

Записываем этого пользователя. И если информационная база не зарегистрирована в системе взаимодействия, то на этом выполнение процедуры прекращается.

Если же база зарегистрирована, то создаем пользователя системы взаимодействия, соответствующего новому пользователю информационной базы. Заполняем имя и полное имя пользователя системы взаимодействия кодом и наименованием нового сотрудника и записываем его.

Таким образом, при записи нового сотрудника в справочник Пользователи будет создан соответствующий ему пользователь информационной базы, а также пользователь системы взаимодействия.

Так как имя пользователя информационной базы и имя пользователя системы взаимодействия у нас хранится в реквизите справочника Код, то перед записью элемента справочника нужно выполнить проверку, чтобы значение реквизита не было пустым. Для этого откроем форму элемента справочника Пользователи, создадим обработчик события формы ПередЗаписью и заполним его следующим образом (листинг 3.9).

Листинг 3.9. Обработчик события «ПередЗаписью» формы элемента справочника «Пользователи»

```

&НаКлиенте
Процедура ПередЗаписью(Отказ, ПараметрыЗаписи)

    Если Объект.Код = "" Тогда
        ПоказатьПредупреждение("Не задано имя пользователя!");
        Отказ = Истина;
    КонецЕсли;

КонецПроцедуры

```

Общение кадровика с сотрудниками в обсуждении «один на один» и в групповом обсуждении

Теперь нам нужно сделать так, чтобы кадровик, нажав на кнопку в форме элемента справочника, мог поздравить принятого сотрудника и познакомить с ним всех остальных пользователей приложения. Для этого при нажатии кнопки Оповестить о принятии нужно создать обсуждение «один на один» кадровика с сотрудником, которого он хочет поздравить с принятием на работу, и групповое обсуждение кадровика со всеми пользователями приложения.

Чтобы реализовать этот сценарий, создадим команду формы ОповеститьОПринятии и поместим ее в командную панель. В свойстве Использование этой команды укажем, что она будет доступна только пользователю с ролью Кадровик. Обработчик этой команды заполним следующим образом (листинг 3.10).

Листинг 3.10. Обработчик команды «ОповеститьОПринятии»

```

&НаКлиенте
Процедура ОповеститьОПринятии(Команда)

    Если Объект.Ссылка.Пустая() Тогда
        ПоказатьПредупреждение("Данные не записаны!");
        Возврат;
    КонецЕсли;

    ОповеститьОПринятииНаСервере();

КонецПроцедуры

```

В этом обработчике вызывается процедура `ОповеститьОПринятииНаСервере()` (листинг 3.11). Это происходит только в том случае, если элемент справочника записан, так как перед выполнением этой процедуры для нового сотрудника уже должны быть созданы соответствующие ему пользователи информационной базы и системы взаимодействия.

Листинг 3.11. Процедура «ОповеститьОПринятииНаСервере»

```
&НаСервере
Процедура ОповеститьОПринятииНаСервере()

    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат;
    КонецЕсли;

    УстановитьПривилегированныйРежим(Истина);

    ИдентификаторПользователяСВКадровик =
        СистемаВзаимодействия.ИдентификаторТекущегоПользователя();
    ПользовательИБСотрудник =
        ПользователиИнформационнойБазы.НайтиПоИмени(СокрЛП(Объект.Код));
    ИдентификаторПользователяСВСотрудник =
        СистемаВзаимодействия.ПолучитьИдентификаторПользователя(
            ПользовательИБСотрудник.УникальныйИдентификатор);
    ПользовательСВСотрудник = СистемаВзаимодействия.ПолучитьПользователя(
        ИдентификаторПользователяСВСотрудник);

    // Поздравление - обсуждение кадровика с сотрудником
    КлючОбсуждения = Строка(ИдентификаторПользователяСВКадровик) + "-" + Строка(
        ИдентификаторПользователяСВСотрудник);
    Обсуждение = СистемаВзаимодействия.ПолучитьОбсуждение(КлючОбсуждения);
    Если Обсуждение = Неопределено Тогда
        Обсуждение = СистемаВзаимодействия.СоздатьОбсуждение();
        Обсуждение.Групповое = Ложь;
        Обсуждение.Ключ = КлючОбсуждения;

        Обсуждение.Участники.Добавить(ИдентификаторПользователяСВКадровик);
        Обсуждение.Участники.Добавить(ИдентификаторПользователяСВСотрудник);
        Обсуждение.Записать();
    КонецЕсли;

    Сообщение = СистемаВзаимодействия.СоздатьСообщение(Обсуждение.Идентификатор);
    Сообщение.Автор = ИдентификаторПользователяСВКадровик;
    Текст = "Поздравляем! С завтрашнего дня Вы приняты на постоянную работу" + ":" + Символы.ПС +
        "В отделение: " + Объект.Подразделение + ", на должность: " + Объект.Должность + Символы.ПС +
        "Дополнительно: " + "http://infoserver/newbie.htm";
    Сообщение.Текст = Новый ФорматированнаяСтрока(Текст);
```



```
// добавить вложение
Поток = ФайловыеПотоки.ОткрытьДляЧтения("с:\поздравление.jpg");
Сообщение.Вложения.Добавить(Поток, "Поздравление", "image/jpeg");
Сообщение.Записать();

// Представление - групповое обсуждение со всеми сотрудниками
КлючОбсуждения = "НовыйСотрудник";
Обсуждение = СистемаВзаимодействия.ПолучитьОбсуждение(КлючОбсуждения);
Если Обсуждение = Неопределено Тогда
    Обсуждение = СистемаВзаимодействия.СоздатьОбсуждение();
    Обсуждение.Заголовок = "Новый сотрудник";
    Обсуждение.Ключ = КлючОбсуждения;

    Обсуждение.Участники.Добавить(
        СистемаВзаимодействия.СтандартныеПользователи.ВсеПользователиПриложения);
    Обсуждение.Записать();
КонецЕсли;

Сообщение = СистемаВзаимодействия.СоздатьСообщение(Обсуждение.Идентификатор);
Сообщение.Автор = ИдентификаторПользователяСВКадровик;
Текст = "Знакомьтесь! У нас появился новый сотрудник" + " :)" + Символы.ПС +
    "С завтрашнего дня принят на постоянную работу" + СокрЛП(Объект.Наименование)
    + Символы.ПС + "В отделение: " + Объект.Подразделение + ", на должность: "
    + Объект.Должность + Символы.ПС + "Электронная почта: "
    + ПользовательСВСотрудник.АдресЭлектроннойПочты;
Сообщение.Текст = Новый ФорматированнаяСтрока(Текст);
Сообщение.Записать();
```

КонецПроцедуры

Поясним текст процедуры.

Сначала нам нужно узнать идентификатор кадровика в системе взаимодействия, а также идентификатор текущего сотрудника (элемента справочника Пользователи), с которым кадровику нужно создать обсуждение.

Идентификатор кадровика получаем как идентификатор текущего пользователя системы взаимодействия, потому что только кадровик может выполнить команду Оповестить о принятии.

Затем мы находим по имени пользователя информационной базы, соответствующего текущему сотруднику. И получаем идентификатор этого пользователя в системе взаимодействия по его уникальному идентификатору в информационной базе.

Теперь создаем обсуждение «один на один» кадровика с сотрудником. Уникальный ключ обсуждения образуем путем сложения строк с идентификаторами этих пользователей в системе взаимодействия.

С помощью метода `ПолучитьОбсуждение()` мы проверяем, существует ли обсуждение с таким ключом. Если нет, то создаем новое обсуждение методом `СоздатьОбсуждение()` и устанавливаем у него свойства `Групповое` (в значение `Ложь`, т.к. по умолчанию оно истинно) и `Ключ`. Заголовок обсуждения «один на один» задавать не надо, так как он автоматически устанавливается как полное имя пользователя системы взаимодействия второго участника обсуждения.

Участников в таком обсуждении может быть всего два: инициатор обсуждения и его собеседник. Добавляем в список участников обсуждения (в коллекцию `Обсуждение.Участники`) идентификаторы кадровика и сотрудника, которые мы получили ранее.

После этого записываем новое обсуждение методом `Записать()`. Теперь нам становится доступен его идентификатор.

Если же обсуждение «один на один» кадровика с сотрудником уже существует, мы получаем его идентификатор как `Обсуждение.Идентификатор`.

Затем с помощью метода `СоздатьСообщение()` мы создаем новое сообщение для обсуждения с этим идентификатором. В свойстве `Автор` запоминаем идентификатор кадровика в системе взаимодействия. В текст сообщения в виде форматированной строки помещаем текст поздравления о приеме на постоянную работу, который кадровик хочет отправить сотруднику.

В текст сообщения добавляем также смайлик («:») и ссылку на сайт с дополнительной информацией. Кроме того, вкладываем в сообщение файл с поздравительной открыткой. Для этого мы сначала создаем файловый поток, в котором файл (картинка

с поздравлением) открывается для чтения. И затем добавляем его в коллекцию вложений (Сообщение.Вложения) нашего сообщения. При этом мы указываем тип содержимого ("image/jpeg"), чтобы система понимала, что при нажатии на ссылку вложения надо открыть картинку.

И в заключение записываем сообщение методом Записать(). Сразу после этого оно будет отправлено сотруднику.

Теперь создаем групповое обсуждение Новый сотрудник кадровика с остальными пользователями приложения.

Сначала мы проверяем, существует ли обсуждение с ключом НовыйСотрудник. Если нет, то мы создаем новое обсуждение и устанавливаем у него свойства Ключ и Заголовок. Затем добавляем в список участников обсуждения сразу всех пользователей системы взаимодействия с помощью свойства СтандартныеПользователи.ВсеПользователиПриложения. И в заключение записываем новое обсуждение.

После этого мы создаем новое сообщение в новом или уже существующем обсуждении Новый сотрудник. В свойстве Автор запоминаем идентификатор кадровика в системе взаимодействия. В текст сообщения в виде форматированной строки помещаем текст представления, с помощью которого кадровик знакомит всех пользователей приложения с новым сотрудником. И записываем сообщение. Сразу после этого оно будет отправлено всем пользователям приложения.

Таким образом, после того как кадровик добавил и записал вновь принятого сотрудника в справочник Пользователи (см. рис. 3.11) и нажал на кнопку Оповестить о принятии, будут созданы два обсуждения: обсуждение «один на один» кадровика с сотрудником, которого он хочет поздравить с принятием на работу (рис. 3.12 сверху), и групповое обсуждение Новый сотрудник кадровика со всеми пользователями приложения для представления нового сотрудника остальным (рис. 3.12 внизу).

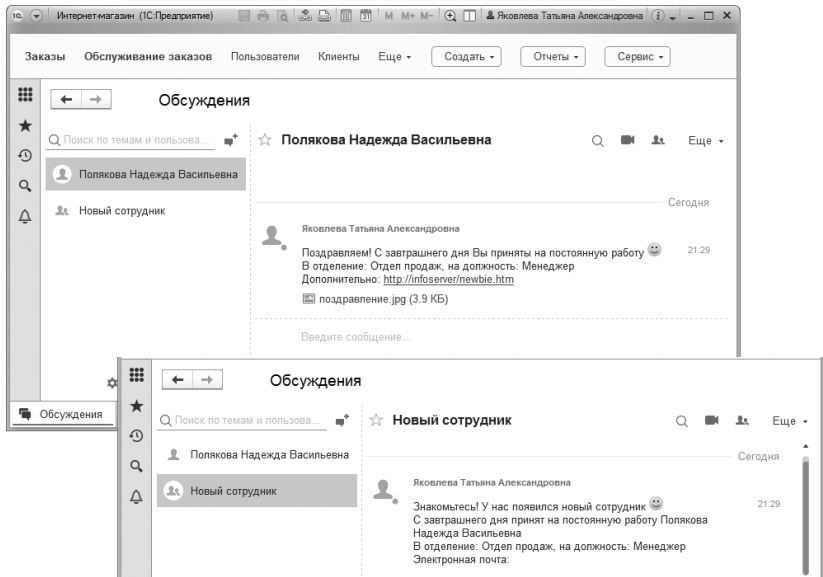


Рис. 3.12. Обсуждение «один на один» кадровика с сотрудником и групповое обсуждение «Новый сотрудник»

Кадровый помощник (бот)

В этом разделе мы рассмотрим, как реализуется смешанное взаимодействие между человеком с одной стороны, и алгоритмом прикладного решения (или «ботом») – с другой. *Бот* (или робот), по определению «Википедии», это программа, которая выполняет действия через интерфейсы, предназначенные для людей.

Мы будем называть нашего бота «Кадровый помощник», но по сути это набор программных алгоритмов, помогающих пользователям решать различные кадровые задачи с помощью системы взаимодействия.

Продолжим рассматривать наш пример с сотрудниками интернет-магазина. Предположим, что организация большая, сотрудников

в ней много и не все друг друга знают. С помощью кадрового помощника мы хотим предоставить любому пользователю информационной базы интернет-магазина возможность быстро найти нужного сотрудника по любому фрагменту его полного имени и получить дополнительную кадровую информацию (должность, подразделение, телефон, электронная почта и т.п.) о выбранном сотруднике, а также распечатать эту информацию.

Кроме того, в зависимости от присутствия сотрудника на работе пользователь должен иметь возможность написать сообщение/позвонить этому сотруднику с помощью системы взаимодействия или отправить ему письмо по электронной почте.

Вообще задач, решаемых при помощи бота, может быть множество, но мы покажем основные из них: как кадровый помощник реагирует на запросы пользователя, на действия пользователя в сообщении, как обрабатывает произвольные команды и т.д.

ПРИМЕЧАНИЕ

Пример реализации смешанного взаимодействия кадрового помощника с сотрудниками интернет-магазина можно посмотреть в конфигурации «КадровыйПомощникИнтернетМагазин», которая прилагается к книге на компакт-диске.

Инициализация кадрового помощника

Начнем с того, что программно, с помощью выполнения общей команды, инициализируем нашего бота. Для этого создадим пользователя информационной базы с именем и соответствующей ролью КадровыйПомощник. Затем создадим всех пользователей системы взаимодействия, соответствующих пользователям информационной базы, включая кадрового помощника. И организуем обсуждение бота «один на один» с каждым из пользователей,

в котором он предлагает свою помощь в поиске информации по сотрудникам, с возможностью вывода этой информации на печать.

Добавим в конфигурацию роль КадровыйПомощник с полными правами. Кроме того, добавим два общих неглобальных модуля: КадровыйПомощник (серверный модуль с установленным флажком Вызов сервера) и КадровыйПомощникКлиент (клиентский модуль). В серверном модуле будут находиться методы, реализующие собственно всю функциональность кадрового помощника, а в клиентском модуле – все обработчики оповещения, с помощью которых бот будет реагировать на различные действия пользователя.

Затем добавим общую команду НачатьРаботуССистемойВзаимодействия, доступную только администратору из группы Сервис. Модуль этой команды заполним следующим образом (листинг 3.12).

Листинг 3.12. Модуль команды «НачатьРаботуССистемойВзаимодействия»

```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)

    ИнициализацияСВ();

    КадровыйПомощникКлиент.ПриНачалеРаботыСистемы();

КонецПроцедуры

&НаСервере
Процедура ИнициализацияСВ()

    КадровыйПомощник.Инициализация();

КонецПроцедуры
```

В обработчике команды вызываются серверная процедура для инициализации системы взаимодействия и процедура для подключения обработчиков оповещения, о которой речь пойдет позже (см. листинг 3.14). Из процедуры для инициализации системы взаимодействия вызывается метод Инициализация(), который мы

поместим в общем модуле КадровыйПомощник и заполним следующим образом (листинг 3.13).

Листинг 3.13. Процедура «Инициализация»

Процедура Инициализация() Экспорт

```
Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
    Возврат;
КонецЕсли;
```

```
УстановитьПривилегированныйРежим(Истина);
```

```
// Создаем пользователя информационной базы КадровыйПомощник
ПользовательИБПомощник = ПользователиИнформационнойБазы.НайтиПоИмени(
    "КадровыйПомощник");
```

```
Если ПользовательИБПомощник = Неопределено Тогда
```

```
    ПользовательИБПомощник = ПользователиИнформационнойБазы.СоздатьПользователя();
    ПользовательИБПомощник.Имя = "КадровыйПомощник";
    ПользовательИБПомощник.ПолноеИмя = "Кадровый помощник";
    ПользовательИБПомощник.ПоказыватьВСпискеВыбора = Ложь;
    ПользовательИБПомощник.Роли.Добавить(Метаданные.Роли.КадровыйПомощник);
    ПользовательИБПомощник.Записать();
```

```
    ПользовательСВПомощник =
        СистемаВзаимодействия.СоздатьПользователя(ПользовательИБПомощник);
    ПользовательСВПомощник.Записать();
```

```
КонецЕсли;
```

```
ИдентификаторПользователяСВПомощник =
    СистемаВзаимодействия.ПолучитьИдентификаторПользователя(
        ПользовательИБПомощник.УникальныйИдентификатор);
```

```
// Создаем пользователей системы взаимодействия
```

```
Для Каждого ПользовательИБ ИЗ ПользователиИнформационнойБазы.ПолучитьПользователей() Цикл
```

```
    Попытка
```

```
        ИдентификаторПользователяСВ =
            СистемаВзаимодействия.ПолучитьИдентификаторПользователя(
                ПользовательИБ.УникальныйИдентификатор);
        ПользовательСВ = СистемаВзаимодействия.ПолучитьПользователя(
            ИдентификаторПользователяСВ);
```

```
    Исключение
```

```
        ПользовательСВ = СистемаВзаимодействия.СоздатьПользователя(ПользовательИБ);
        ПользовательСВ.Записать();
```

КонецПопытки;

ИдентификаторПользователяСВСотрудник = ПользовательСВ.Идентификатор;
Если ИдентификаторПользователяСВСотрудник <> ИдентификаторПользователяСВПомощник
Тогда

```
// создаем обсуждение кадрового помощника с каждым пользователем СВ
КлючОбсуждения = Строка(ИдентификаторПользователяСВПомощник) + " - "
+ Строка(ИдентификаторПользователяСВСотрудник);
Обсуждение = СистемаВзаимодействия.ПолучитьОбсуждение(КлючОбсуждения);
```

```
Если Обсуждение = Неопределено Тогда
Обсуждение = СистемаВзаимодействия.СоздатьОбсуждение();
Обсуждение.Групповое = Ложь;
Обсуждение.Ключ = КлючОбсуждения;
```

```
Обсуждение.Участники.Добавить(ИдентификаторПользователяСВПомощник);
Обсуждение.Участники.Добавить(ИдентификаторПользователяСВСотрудник);
```

```
Обсуждение.Записать();
КонецЕсли;
```

```
Сообщение = СистемаВзаимодействия.СоздатьСообщение(Обсуждение.Идентификатор);
Сообщение.Автор = ИдентификаторПользователяСВПомощник;
Текст = "Приветствую! Я кадровый помощник :)" + Символы.ПС +
"Чтобы получить информацию о "
+ "сотруднике, введите его ФИО полностью или частично.";
Сообщение.Текст = Новый ФорматированнаяСтрока(Текст);
Сообщение.Записать();
```

КонецЕсли;

КонецЦикла;

КонецПроцедуры

В этой процедуре мы сначала проверяем, есть ли в информационной базе пользователь с именем КадровыйПомощник. Если нет, создаем его, а также создаем соответствующего ему пользователя системы взаимодействия. И получаем идентификатор кадрового помощника в системе взаимодействия по его уникальному идентификатору в информационной базе.

Затем обходим всех пользователей информационной базы в цикле и для каждого из них пытаемся получить идентификатор пользователя в системе взаимодействия. Если пользователь инфор-

мационной базы еще не создавался в системе взаимодействия (ни интерактивно, ни программно), будет сгенерировано исключение. В этом случае мы создаем пользователя системы взаимодействия на основании пользователя информационной базы. И после записи также получаем его идентификатор.

Теперь, по мере обхода списка пользователей, нам нужно создать обсуждения «один на один» каждого пользователя с кадровым помощником.

Путем сравнения идентификатора только что созданного пользователя системы взаимодействия и идентификатора кадрового помощника исключаем из списка пользователей бота, так как совершенно не нужно и даже ошибочно создавать обсуждения бота с самим собой.

Уникальный ключ обсуждения образуем путем сложения строк с идентификатором только что созданного пользователя системы взаимодействия и идентификатором бота.

Сначала мы проверяем, существует ли обсуждение с таким ключом. Если нет, то создаем новое обсуждение и устанавливаем у него свойства Групповое (в значение Ложь, т.к. по умолчанию оно истинно) и Ключ. Заголовок обсуждения «один на один» задавать не надо, так как он автоматически устанавливается как полное имя пользователя системы взаимодействия второго участника обсуждения.

Участников в таком обсуждении может быть всего два: инициатор обсуждения и его собеседник. Добавляем в список участников обсуждения идентификаторы кадрового помощника и пользователя, которые мы получили ранее. И в заключение записываем новое обсуждение.

После этого мы создаем новое сообщение в новом или уже существующем обсуждении бота с пользователем системы взаимодействия. В свойстве Автор запоминаем идентификатор бота

в системе взаимодействия. В текст сообщения в виде форматированной строки помещаем приветствие кадрового помощника с приглашением ввести часть имени для поиска подходящих сотрудников. И в заключение записываем сообщение.

В результате при выполнении команды Начать работу с системой взаимодействия кадровый помощник посылает каждому из пользователей приложения приветственное сообщение и предлагает свою помощь в поиске сотрудников (рис. 3.13).

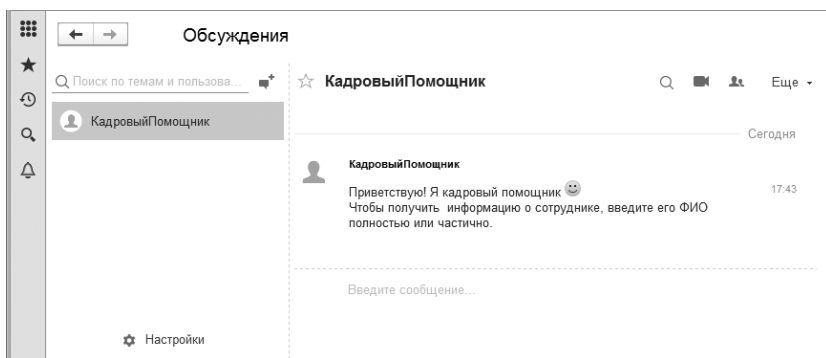


Рис. 3.13. Обсуждение «один на один» кадрового помощника с каждым из сотрудников

Обработка отправки сообщения

После выдачи своего приглашения кадровый помощник «ждет», и как только кто-то из пользователей введет любые символы в ответном сообщении в созданном выше обсуждении «один на один», бот должен найти сотрудников (из справочника Пользователи), полное имя которых (реквизит справочника Наименование) содержит данный фрагмент. И предложить сотруднику на выбор найденные варианты.

Чтобы это реализовать, мы должны при начале работы системы подключить процедуру-обработчик, вызываемую после интерактивной отправки сообщения в обсуждение. Для этого нам нужно описать эту процедуру в клиентском модуле с помощью описания

оповещения и подключить ее методом ПодключитьОбработчикПослеОтправкиСообщения менеджера системы взаимодействия.

Итак, в модуле КадровыйПомощникКлиент поместим экспортную процедуру ПриНачалеРаботыСистемы (листинг 3.14), которую затем будем вызывать из модуля приложения (см. листинг 3.15).

Листинг 3.14. Процедура «ПриНачалеРаботыСистемы»
общего модуля «КадровыйПомощникКлиент»

```
Процедура ПриНачалеРаботыСистемы() Экспорт
```

```
    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат;
    КонечЕсли;
```

```
    Обработчик = Новый ОписаниеОповещения("ОбработкаОтправкиСообщения", ЭтотОбъект);
    СистемаВзаимодействия.ПодключитьОбработчикПослеОтправкиСообщения(Обработчик);
```

```
КонечПроцедуры
```

В этой процедуре мы описываем обработчик оповещения и подключаем процедуру ОбработкаОтправкиСообщения(), которая будет вызываться после интерактивной отправки сообщения в обсуждение.

Затем подключаем этот обработчик при начале работы приложения (листинг 3.15).

Листинг 3.15. Процедура «ПриНачалеРаботыСистемы» модуля приложения

```
Процедура ПриНачалеРаботыСистемы()
```

```
    КадровыйПомощникКлиент.ПриНачалеРаботыСистемы();
```

```
КонечПроцедуры
```

Первым параметром в обработчик оповещения передается копия объекта СообщениеСистемыВзаимодействия, то есть того сообщения, которое было отправлено. Вторым параметром передается копия объекта ОбсуждениеСистемыВзаимодействия, то есть того обсуждения, в которое было послано сообщение.

Экспортную процедуру `ОбработкаОтправкиСообщения()` также поместим в модуле `КадровыйПомощникКлиент` и заполним ее следующим образом (листинг 3.16).

Листинг 3.16. Процедура «`ОбработкаОтправкиСообщения`»

```
Процедура ОбработкаОтправкиСообщения(Сообщение, Обсуждение, ДопПараметры) Экспорт
    КадровыйПомощник.ПолучитьСписокСотрудников(Сообщение.Текст, Обсуждение.Идентификатор);
КонецПроцедуры
```

В этом обработчике мы вызываем процедуру `ПолучитьСписокСотрудников()` и передаем в нее в первом параметре текст сообщения (часть имени сотрудника), введенный пользователем, а во втором – идентификатор обсуждения, которому принадлежит отправленное сообщение.

Процедуру `ПолучитьСписокСотрудников()` поместим в модуле `КадровыйПомощник` и заполним следующим образом (листинг 3.17).

Листинг 3.17. Процедура «`ПолучитьСписокСотрудников`»

```
Процедура ПолучитьСписокСотрудников(ТекстЗапроса, ИдентификаторОбсуждения) Экспорт
    УстановитьПривилегированныйРежим(Истина);

    Сообщение = СистемаВзаимодействия.СоздатьСообщение(ИдентификаторОбсуждения);
    ФИОСотрудника = СокрЛП(ТекстЗапроса);

    Запрос = Новый Запрос;
    Запрос.Текст =
        "ВЫБРАТЬ
        |     Пользователи.Ссылка КАК Ссылка,
        |     Пользователи.Код КАК Код,
        |     Пользователи.Наименование КАК Наименование
        |ИЗ
        |     Справочник.Пользователи КАК Пользователи
        |ГДЕ
        |     Пользователи.Наименование ПОДОБНО """" + &ЧастьНаименования + """"
        |     И Пользователи.ПометкаУдаления = ЛОЖЬ
        |
        |УПОРЯДОЧИТЬ ПО
        |     Наименование";
```

```

Запрос.УстановитьПараметр("ЧастьНаименования", ФИОСотрудника);

РезультатЗапроса = Запрос.Выполнить();
Если РезультатЗапроса.Пустой() Тогда

    Сообщение.Автор = ПолучитьИдентификаторСВПомощника();
    Сообщение.Текст = Новый ФорматированнаяСтрока(
        "По вашему запросу сотрудники не найдены :(");
    Сообщение.Записать();
    Возврат;

КонецЕсли;

Выборка = РезультатЗапроса.Выбрать();
Пока Выборка.Следующий() Цикл
    Если ПолучитьИдентификаторСВСотрудника(Выборка.Ссылка)
        <> СистемаВзаимодействия.ИдентификаторТекущегоПользователя() Тогда

        Сообщение.Действия.Добавить(Выборка.Ссылка, Выборка.Наименование);
    КонецЕсли;
КонецЦикла;

Сообщение.Автор = ПолучитьИдентификаторСВПомощника();
Сообщение.Текст = "Выберите нужного вам сотрудника";
Сообщение.Записать();

КонецПроцедуры

```

В этой процедуре мы сначала создаем сообщение в обсуждении, идентификатор которого передан в процедуру в параметре ИдентификаторОбсуждения.

Затем выполняем запрос, который получает всех сотрудников из справочника Пользователи, наименование которых содержит текст, переданный в параметре ТекстЗапроса.

Если таких сотрудников нет, то выводим соответствующее ответное сообщение бота и на этом прекращаем работу процедуры.

Далее мы обходим выборку из результатов запроса в цикле и добавляем в список действий ответного сообщения бота (Сообщение.Действия) ссылки на найденных сотрудников. При этом мы исключаем из действий сообщения ссылку сотрудника на самого себя.

Идентификатор сотрудника в системе взаимодействия мы определяем по ссылке на текущего сотрудника, полученной при обходе выборки, с помощью функции `ПолучитьИдентификаторСВСотрудника()` (листинг 3.18). А идентификатор автора сообщения, то есть кадрового помощника, мы получаем с помощью функции `ПолучитьИдентификаторСВПомощника()` (листинг 3.19).

Листинг 3.18. Функция «ПолучитьИдентификаторСВСотрудника»

```
Функция ПолучитьИдентификаторСВСотрудника(Ссылка) Экспорт
    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат "";
    КонецЕсли;

    СотрудникКод = СокрЛП(Ссылка.Код);
    ПользовательИБ = ПользователиИнформационнойБазы.НайтиПоИмени(СотрудникКод);
    Если ПользовательИБ.ПолноеИмя = СокрЛП(Ссылка.Наименование) Тогда

        ИдентификаторПользователяСВСотрудник = СистемаВзаимодействия.
            ПолучитьИдентификаторПользователя(ПользовательИБ.УникальныйИдентификатор);
    КонецЕсли;

    Возврат ИдентификаторПользователяСВСотрудник;
КонецФункции
```

В этой функции пользователь информационной базы ищется по имени, хранящемуся в реквизите справочника Код. И если полное имя пользователя совпадает с наименованием сотрудника, то функция возвращает идентификатор сотрудника в системе взаимодействия.

Листинг 3.19. Функция «ПолучитьИдентификаторСВПомощника»

```
Функция ПолучитьИдентификаторСВПомощника() Экспорт
    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат "";
    КонецЕсли;

    ПользовательИБПомощник =
        ПользователиИнформационнойБазы.НайтиПоИмени("КадровыйПомощник");
    Если ПользовательИБПомощник = Неопределено Тогда
        Возврат "";
    КонецЕсли;
```

```
ИдентификаторПользователяСВПомощник =  
СистемаВзаимодействия.ПолучитьИдентификаторПользователя(  
ПользовательИБПомощник.УникальныйИдентификатор);
```

Возврат ИдентификаторПользователяСВПомощник;

КонецФункции

В этой функции пользователь информационной базы ищется по имени бота КадровыйПомощник. И если такой пользователь существует, то функция возвращает его идентификатор в системе взаимодействия.

Таким образом, получив предложение кадрового помощника, любой пользователь приложения может отправить обратно боту часть имени сотрудника, информацию о котором он хотел бы увидеть. В ответ на это бот посылает все найденные совпадения в виде ссылок на сотрудников и просит уточнить свой выбор (рис. 3.14).

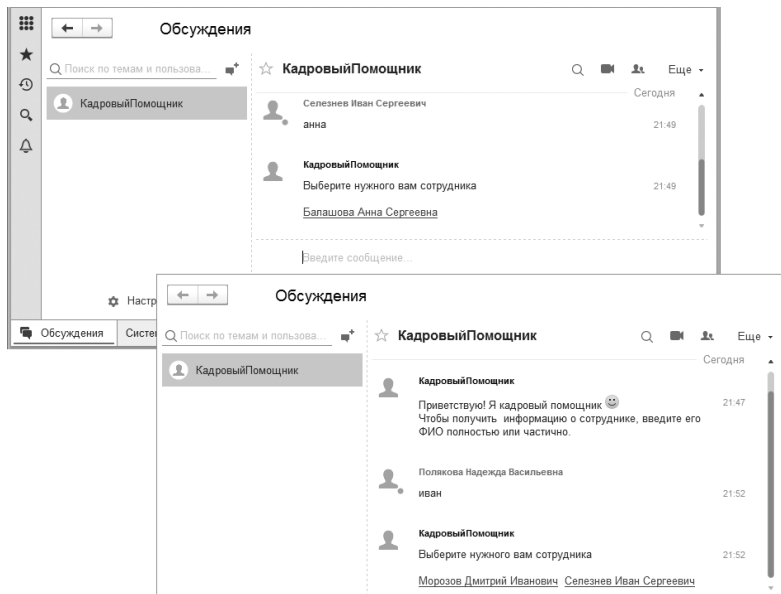


Рис. 3.14. Выбор сотрудника для получения дополнительной информации

Важно понимать, что ссылки на сотрудников – это не навигационные ссылки, нажатие на которые никак не обрабатывается и автоматически приводит к открытию формы объекта. Это гиперссылки действий, нажатие на которые обрабатывается в обработчике действий сообщения, о котором будет рассказано в следующем разделе.

Обработка действий сообщения

Итак, кадровый помощник послал в сообщении пользователю набор ссылок на сотрудников, из которых тот будет выбирать. Эти ссылки на самом деле являются действиями сообщения, которые бот должен обработать. То есть при нажатии пользователя на гиперссылку с именем сотрудника бот должен выдать дополнительную кадровую информацию о выбранном сотруднике.

Про обработку действий сообщения уже рассказывалось подробно в разделе «Обработка действий сообщения», поэтому еще раз напомним, как это делается.

Чтобы реализовать этот сценарий, мы должны при начале работы системы подключить процедуру-обработчик, вызываемую при нажатии на гиперссылку в сообщении, заданную в свойстве сообщения Действия. Для этого нам нужно описать эту процедуру с помощью описания оповещения и подключить ее методом ПодключитьОбработчикДействияСообщения менеджера системы взаимодействия.

В предыдущем разделе мы уже описывали процедуру ПриНачалеРаботыСистемы() в модуле КадровыйПомощникКлиент, которая вызывается из модуля приложения при начале работы системы. Мы использовали эту процедуру для подключения обработчика отправки сообщений.

Теперь дополним эту процедуру следующим образом (листинг 3.20).

Листинг 3.20. Процедура «ПриНачалеРаботыСистемы»
общего модуля «КадровыйПомощникКлиент»

Процедура ПриНачалеРаботыСистемы() Экспорт

Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
 Возврат;
КонецЕсли;

Обработчик = Новый ОписаниеОповещения("ОбработкаОтправкиСообщения", ЭтотОбъект);
СистемаВзаимодействия.ПодключитьОбработчикПослеОтправкиСообщения(Обработчик);

**Обработчик2 = Новый ОписаниеОповещения("ОбработкаДействияСообщения", ЭтотОбъект);
СистемаВзаимодействия.ПодключитьОбработчикДействияСообщения(Обработчик2);**

КонецПроцедуры

В выделенном фрагменте процедуры мы описываем обработчик оповещения и подключаем процедуру `ОбработкаДействияСообщения()`, которая будет вызываться при нажатии гиперссылки (выполнения действия) в сообщении.

Первым параметром в обработчик оповещения передается копия объекта `СообщениеСистемыВзаимодействия`, то есть того сообщения, в котором нажата гиперссылка действия. Вторым параметром передается `Действие` – произвольное значение, заданное для нажатой гиперссылки.

Экспортную процедуру `ОбработкаДействияСообщения()` поместим в модуле `КадровыйПомощникКлиент` и заполним ее следующим образом (листинг 3.21).

Листинг 3.21. Процедура «ОбработкаДействияСообщения»

Процедура ОбработкаДействияСообщения(Сообщение, Действие, ДопПараметры) Экспорт

Если ТипЗнч(Действие) = Тип("СправочникСсылка.Пользователи") Тогда

 КадровыйПомощник.ПолучитьИнформациюОСотруднике(Действие, Сообщение.Обсуждение);
КонецЕсли;

КонецПроцедуры

Как уже говорилось, обработчик действия сообщения будет вызываться при нажатии гиперссылки в любом сообщении. Поэтому сначала мы выполняем проверку, то ли это сообщение, в котором надо показывать дополнительную информацию о сотруднике.

В случае если параметр Действие имеет тип ссылки на справочник Пользователи, мы вызываем процедуру ПолучитьИнформациюОСотруднике() и передаем в нее ссылку на сотрудника и идентификатор обсуждения, в которое кадровый помощник должен поместить дополнительную информацию о сотруднике.

Процедуру ПолучитьИнформациюОСотруднике() поместим в модуле КадровыйПомощник и заполним следующим образом (листинг 3.22).

Листинг 3.22. Процедура «ПолучитьИнформациюОСотруднике»

Процедура ПолучитьИнформациюОСотруднике(СсылкаНаСотрудника, ИдентификаторОбсуждения) Экспорт

```
УстановитьПривилегированныйРежим(Истина);  
Сообщение = СистемаВзаимодействия.СоздатьСообщение(ИдентификаторОбсуждения);
```

```
Запрос = Новый Запрос;  
Запрос.Текст =
```

```
"ВЫБРАТЬ  
|   Пользователи.Код КАК Код,  
|   Пользователи.Наименование КАК Наименование,  
|   ПРЕДСТАВЛЕНИЕ(Пользователи.Подразделение) КАК Подразделение,  
|   ПРЕДСТАВЛЕНИЕ(Пользователи.Должность) КАК Должность  
|ИЗ  
|   Справочник.Пользователи КАК Пользователи  
|ГДЕ  
|   Пользователи.Ссылка = &Ссылка  
|   И Пользователи.ПометкаУдаления = ЛОЖЬ";
```

```
Запрос.УстановитьПараметр("Ссылка", СсылкаНаСотрудника);
```

```
РезультатЗапроса = Запрос.Выполнить();  
Если РезультатЗапроса.Пустой() Тогда
```

```
Сообщение.Автор = ПолучитьИдентификаторСВПомощника();  
Сообщение.Текст = Новый ФорматированнаяСтрока(  
    "По вашему запросу информация не найдена :(");  
Сообщение.Записать();  
Возврат;
```

```

КонецЕсли;

Выборка = РезультатЗапроса.Выбрать();
Выборка.Следующий();
Текст = "Сотрудник: " + Выборка.Наименование + Символы.ПС +
"Подразделение: " + Выборка.Подразделение + Символы.ПС +
"Должность: " + Выборка.Должность;

ДопИнформация = ДополнительнаяИнформацияОСотруднике(СсылкаНаСотрудника);

Текст = Текст + Символы.ПС +
"Телефон: " + ДопИнформация.Телефон + Символы.ПС +
"Эл. почта: " + ДопИнформация.ЭлПочта + Символы.ПС +
"Работает? " + ДопИнформация.СтрокаСтатуса;

Сообщение.Автор = ПолучитьИдентификаторСВПомощника();
Сообщение.Текст = Новый ФорматированнаяСтрока(Текст);

Если ДопИнформация.Статус = 0 Тогда
    Сообщение.Действия.Добавить("Connect", "Отправить сообщение / позвонить");
ИначеЕсли ДопИнформация.Статус > 0 Тогда
    Сообщение.Действия.Добавить("EMail", "Отправить сообщение по эл.почте");
КонецЕсли;

ПечатьИнформации = Новый Структура;
ПечатьИнформации.Вставить("Сотрудник", СсылкаНаСотрудника);
ПечатьИнформации.Вставить("Печатать", Истина);
Сообщение.Данные = ПечатьИнформации;
Сообщение.Записать();

```

КонецПроцедуры

В этой процедуре мы сначала создаем сообщение в обсуждении, идентификатор которого передан в процедуру в параметре ИдентификаторОбсуждения.

Затем выполняем запрос, который получает кадровую информацию из справочника Пользователи, для сотрудника, ссылка на которого содержится в параметре СсылкаНаСотрудника.

Если результат запроса пустой, то выводим соответствующее ответное сообщение бота и на этом прекращаем работу процедуры. В противном случае мы заполняем текст сообщения кадровой информацией сотрудника, полученной запросом.

Теперь представим себе, что дополнительная информация о сотруднике хранится где-то еще. Для получения этой информации мы используем функцию `ДополнительнаяИнформацияОСотруднике()`, которая возвращает структуру с полями `Телефон`, `ЭлПочта`, `Статус` и т. п.

В зависимости от присутствия сотрудника на работе (значение поля структуры `Статус`), мы добавляем в список действий ответного сообщения бота действие "Connect" с представлением «Отправить сообщение/позвонить» или, если сотрудника нет в данный момент на работе, действие "EMail" с представлением «Отправить сообщение по эл. почте».

Идентификатор автора сообщения, то есть кадрового помощника, в системе взаимодействия мы получаем с помощью функции `ПолучитьИдентификаторСВПомощника()` (см. листинг 3.19).

В данных сообщения (`Сообщение.Данные`) мы передаем структуру `ПечатьИнформации` с полями `Сотрудник` (ссылка на сотрудника) и `Печатать` (признак печати). Ссылка на сотрудника нам нужна для обработки действий сообщения (см. листинг 3.23).

А также значения обоих полей структуры `ПечатьИнформации` нам нужны потому, что мы хотим предоставить пользователю возможность распечатать полученную от бота кадровую информацию. Эти поля нам понадобятся при обработке формирования команд в сообщении, о которой речь пойдет в следующем разделе «Обработка формирования команд в сообщении» на стр. 104.

В результате, после того как пользователь выберет нужного ему сотрудника, кадровый помощник найдет и покажет дополнительную информацию об этом сотруднике. При этом если сотрудник в данный момент на работе, то, нажав на гиперссылку, пользователь должен иметь возможность отправить ему сообщение или позвонить в обсуждении. В противном случае пользователь должен иметь возможность отправить ему письмо по электронной почте (рис. 3.15).

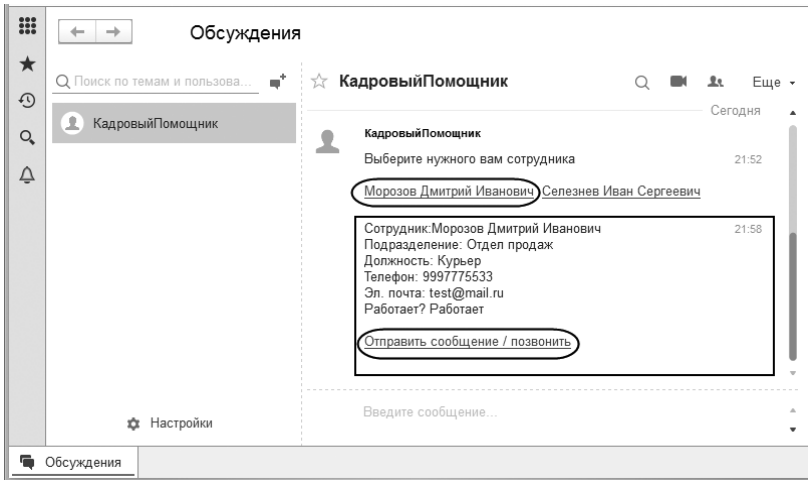


Рис. 3.15. Дополнительная информация о сотруднике

Чтобы реализовать этот сценарий, нам нужно дополнить обработчик оповещения `ОбработкаДействияСообщения()` в модуле `КадровыйПомощникКлиент` следующим образом (листинг 3.23).

Листинг 3.23. Процедура «ОбработкаДействияСообщения»

Процедура `ОбработкаДействияСообщения(Сообщение, Действие, ДопПараметры)` Экспорт

Если `ТипЗнч(Действие) = Тип("СправочникСсылка.Пользователи")` Тогда

`КадровыйПомощник.ПолучитьИнформациюОСотруднике(Действие, Сообщение.Обсуждение);`
 КонецЕсли;

Если `Действие = "Connect"` Тогда

`СсылкаНаОбсуждение = КадровыйПомощник.ОтправитьСообщениеСотруднику(`
 `Сообщение.Данные.Сотрудник);`

`ПерейтиПоНавигационнойСсылке(СсылкаНаОбсуждение);`

КонецЕсли;

Если `Действие = "EMail"` Тогда

`КадровыйПомощник.НаписатьПисьмоСотруднику(Сообщение.Данные.Сотрудник);`

КонецЕсли;

КонецПроцедуры

В выделенном фрагменте обработчика, в случае если параметр Действие имеет значение "Connect", мы вызываем функцию ОтправитьСообщениеСотруднику() и в параметре Сообщение. Данные.Сотрудник передаем в нее ссылку на сотрудника, с которым нужно организовать обсуждение.

Если параметр Действие имеет значение "EMail", мы вызываем функцию НаписатьПисьмоСотруднику(), на которой мы не будем подробно останавливаться.

Функцию ОтправитьСообщениеСотруднику() поместим в модуле КадровыйПомощник и заполним следующим образом (листинг 3.24).

Листинг 3.24. Функция «ОтправитьСообщениеСотруднику»

```
Функция ОтправитьСообщениеСотруднику(Сотрудник) Экспорт
    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат "";
    КонецЕсли;

    УстановитьПривилегированныйРежим(Истина);

    ИдентификаторПользователяСВСотрудник = ПолучитьИдентификаторСВСотрудника(Сотрудник);

    // создаем обсуждение текущего пользователем СВ с выбранным сотрудником
    ИдентификаторТекущегоПользователяСВ =
        СистемаВзаимодействия.ИдентификаторТекущегоПользователя();
    КлючОбсуждения = Строка(ИдентификаторТекущегоПользователяСВ) + "-" +
        Строка(ИдентификаторПользователяСВСотрудник);
    Обсуждение = СистемаВзаимодействия.ПолучитьОбсуждение(КлючОбсуждения);

    Если Обсуждение = Неопределено Тогда
        Обсуждение = СистемаВзаимодействия.СоздатьОбсуждение();
        Обсуждение.Групповое = Ложь;
        Обсуждение.Ключ = КлючОбсуждения;

        Обсуждение.Участники.Добавить(ИдентификаторТекущегоПользователяСВ);
        Обсуждение.Участники.Добавить(ИдентификаторПользователяСВСотрудник);

        Обсуждение.Записать();
    КонецЕсли;

    Возврат ПолучитьНавигационнуюСсылку(Обсуждение.Идентификатор);

КонецФункции
```

В этой функции, в случае если оно еще не существует, создается обсуждение «один на один» текущего пользователя системы взаимодействия с сотрудником, о котором бот показал дополнительную информацию (см. рис. 3.15).

Создание подобного обсуждения уже рассматривалось и объяснялось нами подробно, когда мы программно создавали обсуждение кадровика с вновь принятым сотрудником (см. процедуру `ОповеститьОПринятииНаСервере()`, листинг 3.11), а также когда мы создавали обсуждения кадрового помощника с каждым из пользователей приложения (см. процедуру `Инициализация()`, листинг 3.13). Поэтому мы не будем еще раз комментировать эту функцию.

Заметим лишь, что функция возвращает навигационную ссылку на уже существующее или вновь созданное обсуждение, по которой затем происходит переход к этому обсуждению (см. процедуру `ОбработкаДействияСообщения()`, листинг 3.23).

Таким образом, после того как пользователь нажмет гиперссылку `Отправить сообщение / позвонить` в сообщении с дополнительной информацией о сотруднике, полученной от кадрового помощника, будет создано (или найдено существующее) обсуждение «один на один» пользователя с этим сотрудником. И пользователь сразу же попадет в это обсуждение (рис. 3.16).

При этом, как мы видим на рис. 3.16, пользователь может распечатать эту информацию с помощью произвольной команды `Печать`. О том, как это делается, будет рассказано в следующем разделе.

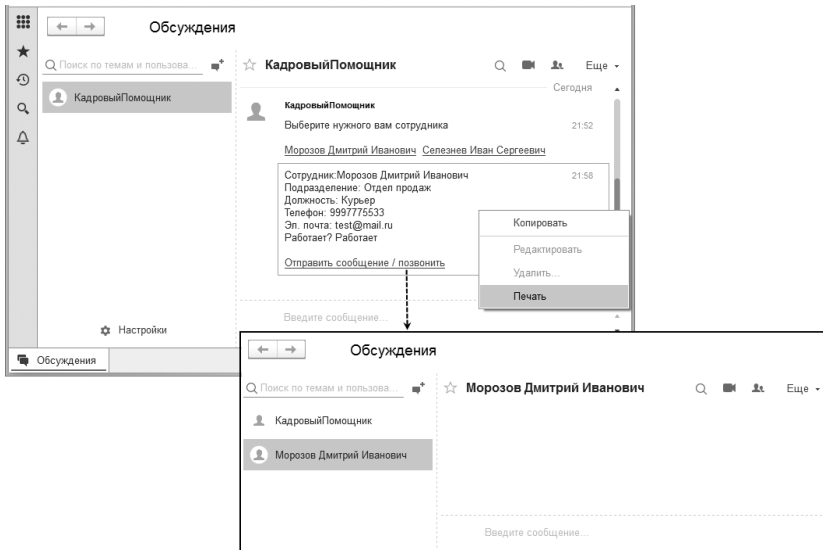


Рис. 3.16. Создание обсуждения с сотрудником

Обработка формирования команд в сообщении

Кроме того, что бот отправляет пользователю дополнительную кадровую информацию о сотруднике, он предоставляет ему возможность распечатать эту информацию с помощью команды Печать, добавленной в контекстное меню сообщения.

Печать – это не стандартная команда сообщения (наподобие Копировать, Удалить и т.п.), а наша собственная команда, которую мы должны добавить в список команд сообщения и обеспечить выполнение этой команды.

Для этого мы должны при начале работы системы подключить процедуру-обработчик, вызываемую при нажатии правой кнопки мыши на сообщении. Для этого нам нужно описать эту процедуру с помощью описания оповещения и подключить ее методом ПодключитьОбработчикФормированияКоманд менеджера системы взаимодействия.

Ранее мы уже описывали процедуру `ПриНачалеРаботыСистемы()` в модуле `КадровыйПомощникКлиент`, которая вызывается из модуля приложения при начале работы системы. Мы использовали эту процедуру для подключения обработчика отправки сообщений и обработчика действий сообщения.

Теперь дополним эту процедуру следующим образом (листинг 3.25).

Листинг 3.25. Процедура «`ПриНачалеРаботыСистемы`» общего модуля «`КадровыйПомощникКлиент`»

Процедура `ПриНачалеРаботыСистемы()` Экспорт

```
Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
    Возврат;
КонецЕсли;
```

```
Обработчик = Новый ОписаниеОповещения("ОбработкаОтправкиСообщения", ЭтотОбъект);
СистемаВзаимодействия.ПодключитьОбработчикПослеОтправкиСообщения(Обработчик);
```

```
Обработчик2 = Новый ОписаниеОповещения("ОбработкаДействияСообщения", ЭтотОбъект);
СистемаВзаимодействия.ПодключитьОбработчикДействияСообщения(Обработчик2);
```

```
Обработчик3 = Новый ОписаниеОповещения("ОбработкаФормированияКоманд", ЭтотОбъект);
СистемаВзаимодействия.ПодключитьОбработчикФормированияКоманд(Обработчик3);
```

КонецПроцедуры

В выделенном фрагменте процедуры мы описываем обработчик оповещения и подключаем процедуру `ОбработкаФормированияКоманд()`, которая будет вызываться при нажатии правой кнопки мыши в сообщении.

Первым параметром в обработчик оповещения передается объект `ПараметрыФормированияКомандСистемыВзаимодействия`, в свойстве `Сообщение` которого содержатся данные (`Параметры.Сообщение.Данные`), нужные нам для формирования нашей команды.

Вторым параметром передается массив стандартных команд – элементов типа `ОписаниеКомандыСистемыВзаимодействия`.

При вызове обработчика массив заполняется списком команд по умолчанию. Этот массив можно дополнить своими собственными командами.

Важно понимать, что обработчик формирования команд будет вызываться в любом сообщении не только при нажатии правой кнопки мыши, но и при нажатии левой или правой кнопки мыши на гиперссылке в тексте сообщения, на вложении, а также на гиперссылке действия.

Поэтому для работы нашего алгоритма нам потребуется проверка – то ли это сообщение, в котором нужно показывать команду Печать.

Экспортную процедуру `ОбработкаФормированияКоманд()` поместим в модуле `КадровыйПомощникКлиент` и заполним ее следующим образом (листинг 3.26).

Листинг 3.26. Процедура «`ОбработкаФормированияКоманд`»

```
Процедура ОбработкаФормированияКоманд(
    Параметры, Команды, КомандаПоУмолчанию, ДопПараметры) Экспорт

    ДанныеСообщения = Параметры.Сообщение.Данные;
    Если ТипЗнч(ДанныеСообщения) = Тип("Структура") И ДанныеСообщения.Свойство("Печатать") Тогда

        ОбработчикКоманды = Новый ОписаниеОповещения(
            "ОбработкаВыполненияКоманды", ЭтотОбъект, ДанныеСообщения.Сотрудник);
        НоваяКоманда = Новый ОписаниеКомандыСистемыВзаимодействия(
            ОбработчикКоманды, "Печать");
        НоваяКоманда.Доступность = ДанныеСообщения.Печатать;
        Команды.Добавить(НоваяКоманда);
    КонецЕсли;

КонецПроцедуры
```

В этом обработчике мы сначала проверяем, надо ли в контекстном меню этого сообщения показывать команду Печать.

Напомним, что при создании сообщения с дополнительной информацией о сотруднике (см. листинг 3.22) мы сохранили в данных

сообщения структуру `ПечатьИнформации` с полями `Сотрудник` (ссылка на сотрудника) и `Печатать` (признак печати).

И теперь при нажатии правой кнопки мыши на сообщении, как только сработает обработчик формирования команд, мы получим данные сообщения в значении первого параметра `Параметры.Сообщение.Данные`.

Поскольку, как уже говорилось, обработчик формирования команд срабатывает при нажатии мыши на любом сообщении, мы проверяем, есть ли в этой структуре данных поле `Печатать`. Если есть, значит, в контекстном меню этого сообщения нужно показывать команду `Печать`.

В этом случае мы описываем процедуру-обработчик оповещения, которая будет выполняться при вызове нашей команды. Третьим параметром в эту процедуру мы передаем ссылку на сотрудника (значение поля структуры данных сообщения `Сотрудник`), информацию о котором нужно распечатать.

Затем создаем нашу новую команду конструктором объекта `ОписаниеКомандыСистемыВзаимодействия`, указав обработчик команды и ее имя – `Печать`. Устанавливаем доступность нашей команды в зависимости от значения поля структуры данных сообщения `Печатать`.

После этого просто добавляем ее как новый элемент массива команд, содержащегося во втором параметре `Команды` обработчика формирования команд.

Экспортную процедуру `ОбработкаВыполненияКоманды()` поместим в модуле `КадровыйПомощникКлиент` и заполним ее следующим образом (листинг 3.27).

Листинг 3.27. Процедура «ОбработкаВыполненияКоманды»

Процедура ОбработкаВыполненияКоманды(ДопПараметры) Экспорт

КадровыйПомощник.ПечатьИнформацииОСотруднике(ДопПараметры);

КонецПроцедуры

В обработчике команды мы вызываем процедуру ПечатьИнформацииОСотруднике() модуля КадровыйПомощник и передаем туда ссылку на сотрудника, которая содержится в параметре обработчика ДопПараметры.

Таким образом, при вызове команды Печать из контекстного меню сообщения, полученного от кадрового помощника, работает наш обработчик ОбработкаВыполненияКоманды(). И в результате дополнительная кадровая информация о сотруднике может быть распечатана (рис. 3.17).

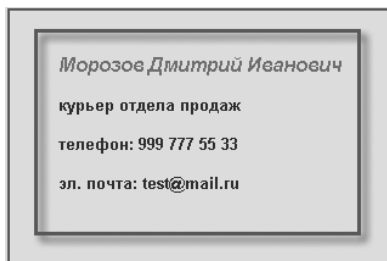


Рис. 3.17. Печать кадровой информации о сотруднике

ГЛАВА 4

ПРОГРАММНОЕ ВЗАИМОДЕЙСТВИЕ

В этой главе мы рассмотрим, как можно использовать систему взаимодействия для организации программного взаимодействия различных частей прикладного решения друг с другом. А именно – как организовать передачу информации с сервера в клиентское приложение.

Дело в том, что с помощью «обычных» программных механизмов можно передавать управление только в одну сторону – с клиента на сервер, но не наоборот. Например, когда с клиента запускается какая-то длительная операция, то клиенту остается только ждать, когда ее выполнение закончится. До этого момента «понять», делается ли что-то на сервере и сколько времени еще осталось

до завершения, нельзя. Поэтому клиенту приходится постоянно опрашивать сервер на эту тему.

Но теперь с помощью системы взаимодействия можно передавать информацию с сервера на клиент. Например, по мере выполнения фонового задания сервер может посылать клиенту сообщения, в которых содержится информация о ходе выполнения задания, запущенного на этом клиенте. Делается это в специальном служебном обсуждении, на которое подписывается клиент. После этого он будет получать в этом служебном обсуждении сообщения и соответствующим образом обрабатывать их. Например, показывать пользователю прогресс выполнения длительной операции.

Таким образом, пользователь, запустивший сложную обработку или какой-то регламентированный отчет, сможет визуально оценить, сколько времени осталось до завершения операции, не нервничать, а сделать «кофейную паузу», например.

Кроме того, передачу информации с сервера в клиентское приложение можно использовать в самых разных ситуациях. Например:

- для отображения прогресса длительной серверной операции;
- для уведомления пользователей о перезагрузке сервера и принудительного завершения клиентских приложений;
- для уведомления пользователя о входящем SIP-звонке;
- для поддержки прохождения бизнес-процессов;
- для реализации «напоминалок», уведомлений и пр.

Поясним сказанное на небольших примерах. В первом примере мы рассмотрим очень востребованную задачу взаимодействия клиента и сервера на предмет информирования клиента о ходе выполнения длительных операций на сервере. Во втором примере рассмотрим отправку уведомлений в клиентское приложение при наступлении каких-то событий на сервере.

Передача с сервера на клиент информации о ходе выполнения длительной операции

Продолжим, как и в предыдущей главе, рассматривать работу кадровика в организации с большим количеством сотрудников. Формирование некоторых регламентированных отчетов может занять продолжительное время, поэтому кадровику было бы очень удобно видеть прогресс формирования таких отчетов в панели состояния.

Чтобы это реализовать, нужно создать неотображаемое служебное обсуждение, участником которого будет наш кадровик. Ключ этого обсуждения должен быть уникальным и однозначно идентифицировать кадровика – например, по его идентификатору в системе взаимодействия. Создавать служебное обсуждение нужно только один раз. Можно это сделать в обработке в момент программного создания пользователей системы взаимодействия.

Затем каждый раз при запуске клиентского приложения нужно подписываться на это служебное обсуждение, предназначенное кадровику. Это делается с помощью метода менеджера системы взаимодействия `НачатьПодключениеОбработчикаНовыхСообщений()`. Этот метод подключает к обсуждению с указанным ключом (к служебному обсуждению кадровика) обработчик, который будет вызван после появления новых сообщений в обсуждении.

В результате этот обработчик будет срабатывать всякий раз, когда с сервера в служебное обсуждение будут посылаться сообщения, в данных которых содержится информация о ходе выполнения фонового задания. В этом обработчике, в зависимости от данных, переданных в сообщениях с сервера, и нужно выполнять необходимые действия – например, показывать прогресс выполнения длительной операции.

ПРИМЕЧАНИЕ

Пример реализации программного взаимодействия клиента (кадровика) с сервером можно посмотреть в конфигурации «ПрограммноеВзаимодействиеКлиентСервер», которая прилагается к книге на компакт-диске.

Итак, сначала создадим служебное обсуждение для получения информации с сервера, участником которого будет кадровик. Для упрощения примера будем реализовывать соответствующий код с помощью общей команды `СозданиеСлужебногоОбсуждения`, которую нужно вызвать всего один раз. Однако (как видно из кода команды), даже если выполнить команду несколько раз, ничего «страшного» не произойдет.

Добавим эту команду и заполним ее модуль следующим образом (листинг 4.1).

Листинг 4.1. Процедура «ОбработкаКоманды»

```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)

    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат;
    КонецЕсли;

    КлючСлужебногоОбсуждения = СлужебныеСообщенияКлиент.ПолучитьКлючСлужебногоОбсуждения();
    СоздатьСлужебноеОбсуждение(КлючСлужебногоОбсуждения);

КонецПроцедуры

&НаСервере
Процедура СоздатьСлужебноеОбсуждение(КлючСлужебногоОбсуждения)

    Если СистемаВзаимодействия.ПолучитьОбсуждение(КлючСлужебногоОбсуждения) = Неопределено Тогда
        СлужебноеОбсуждение = СистемаВзаимодействия.СоздатьОбсуждение();
        СлужебноеОбсуждение.Отображаемое = Ложь;
        СлужебноеОбсуждение.Ключ = КлючСлужебногоОбсуждения;
        СлужебноеОбсуждение.Участники.Добавить(
            СистемаВзаимодействия.ИдентификаторТекущегоПользователя());
        СлужебноеОбсуждение.Записать();
    КонецЕсли;

КонецПроцедуры
```


В обработчике этой команды с помощью функции `ПолучитьКлючСлужебногоОбсуждения()` общего модуля `СлужебныеСообщенияКлиент` мы получаем ключ служебного обсуждения как произвольную строку, содержащую идентификатор кадровика в системе взаимодействия (см. листинг 4.2).

Затем в процедуре `СоздатьСлужебноеОбсуждение()` мы создаем обсуждение с ключом `КлючСлужебногоОбсуждения`, если обсуждение с таким ключом еще не существует. Добавляем текущего пользователя системы взаимодействия (то есть кадровика) в качестве единственного участника этого обсуждения. Чтобы обсуждение не отображалось в интерфейсе кадровика, мы устанавливаем свойство обсуждения `Отображаемое` в значение `Ложь`.

Функцию `ПолучитьКлючСлужебногоОбсуждения()` мы поместим в общем клиентском модуле `СлужебныеСообщенияКлиент` (листинг 4.2).

Листинг 4.2. Функция «ПолучитьКлючСлужебногоОбсуждения»

```
Функция ПолучитьКлючСлужебногоОбсуждения() Экспорт
    Возврат "Сообщения сервера для кадровика"
        + СистемаВзаимодействия.ИдентификаторТекущегоПользователя();
КонецФункции
```

Затем добавим еще одну общую команду `ПодпискаНаСлужебноеОбсуждение`. В обработчике этой команды мы реализуем подписку на служебное обсуждение с ключом `КлючСлужебногоОбсуждения` (листинг 4.3).

Листинг 4.3. Процедура «ОбработкаКоманды»

```
&НаКлиенте
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)
    Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда
        Возврат;
    КонецЕсли;
    КлючСлужебногоОбсуждения = СлужебныеСообщенияКлиент.ПолучитьКлючСлужебногоОбсуждения();
```

```
ОбработкаСообщенийСервера = Новый ОписаниеОповещения(  
    "ОбработкаСообщенийСервера", СлужбныеСообщенияКлиент);  
СистемаВзаимодействия.НачатьПодключениеОбработчикаНовыхСообщений(  
    КлючСлужебногоОбсуждения, ОбработкаСообщенийСервера);
```

КонецПроцедуры

В обработчике команды мы сначала получаем ключ служебного обсуждения. Затем вызываем метод менеджера системы взаимодействия `НачатьПодключениеОбработчикаНовыхСообщений()`.

Первый параметр метода – описание процедуры, которая будет вызвана после выполнения подключения обработчика, мы опускаем. Во втором параметре метода мы передаем ключ служебного обсуждения. В третьем параметре мы передаем описание обработчика оповещения – процедуры `ОбработкаСообщенийСервера()`, которая будет вызвана после появления новых сообщений в обсуждении с ключом `КлючСлужебногоОбсуждения`.

Саму процедуру-обработчик мы поместим в общем клиентском модуле `СлужбныеСообщенияКлиент` (листинг 4.4). Первым параметром в обработчик оповещения передается копия объекта `СообщениеСистемыВзаимодействия`, то есть того сообщения, которое было получено в служебном обсуждении.

Листинг 4.4. Процедура «ОбработкаСообщенийСервера»

```
Процедура ОбработкаСообщенийСервера(Сообщение, ДопПараметры) Экспорт
```

```
    ИндексЗадания = глИдентификаторыФоновыхЗаданий.Найти(  
        Сообщение.Данные.ИдентификаторЗадания);  
    Если НЕ ИндексЗадания = Неопределено Тогда  
        Если Сообщение.Данные.СпособОбработки = "Индикатор" Тогда  
            Состояние("Выполняется обработка данных", Сообщение.Данные.Значение);  
        ИначеЕсли Сообщение.Данные.СпособОбработки = "НеОтслеживать" Тогда  
            глИдентификаторыФоновыхЗаданий.Удалить(ИндексЗадания);  
        КонецЕсли;
```

```
    КонецЕсли;
```

```
КонецПроцедуры
```

Таким образом, при получении новых сообщений в нашем служебном обсуждении будет выполняться процедура `ОбработкаСообщенийСервера()`, в которой и производится обработка данных, содержащихся в сообщениях.

Саму процедуру пока комментировать не будем. Мы вернемся к этому позже, после того как рассмотрим процедуры, в которых эти сообщения отправляются с сервера.

Далее нам нужно запустить длительную операцию, которая должна исполняться на сервере, как фоновое задание. Каждое фоновое задание исполняется в отдельном системном сеансе. В результате после запуска фонового задания на клиенте пользовательский сеанс может выполнять другие действия, например обрабатывать сообщения, появляющиеся в служебном обсуждении.

При этом необходимо учитывать, что от имени одного и того же пользователя может быть одновременно запущено несколько клиентских приложений, каждое из которых подписано на служебное обсуждение этого пользователя. Значит, все они отреагируют на сообщение, которое появится в этом обсуждении. А нам нужно, чтобы клиент обрабатывал только «свои» фоновые задания.

Поэтому при запуске фонового задания нужно снабдить его уникальным идентификатором, который нужно передать в само фоновое задание, и кроме этого вернуть на клиент. Тогда клиент будет знать «свои» фоновые задания, а фоновое задание, посылая сообщение, сможет однозначно себя идентифицировать.

Для хранения идентификаторов заданий в модуле приложения объявим глобальный массив `гИдентификаторыФоновыхЗаданий` (листинг 4.5).

Листинг 4.5. Модуль приложения

```
Перем глИдентификаторыФоновыхЗаданий Экспорт;  
глИдентификаторыФоновыхЗаданий = Новый Массив;
```

Запускать фоновое задание мы будем с помощью еще одной общей команды ЗапуститьДлительнуюОперацию. Обработчик этой команды заполним следующим образом (листинг 4.6).

Листинг 4.6. Процедура «ОбработкаКоманды»

```
&НаКлиенте  
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)  
  
    КлючСлужебногоОбсуждения = СлужебныеСообщенияКлиент.ПолучитьКлючСлужебногоОбсуждения();  
    ИдентификаторЗадания = СлужебныеСообщения.ОбработатьФоновымЗаданием(  
        КлючСлужебногоОбсуждения);  
  
    глИдентификаторыФоновыхЗаданий.Добавить(ИдентификаторЗадания);  
  
КонецПроцедуры
```

В обработчике команды мы сначала получаем ключ служебного обсуждения. Затем вызываем функцию ОбработатьФоновымЗаданием() общего модуля СлужебныеСообщения и передаем в нее этот ключ.

Эта функция возвращает идентификатор фонового задания, присвоенный ему перед запуском. Этот идентификатор мы запоминаем в глобальном массиве глИдентификаторыФоновыхЗаданий.

Функцию ОбработатьФоновымЗаданием() мы поместим в общем серверном модуле СлужебныеСообщения и заполним ее следующим образом (листинг 4.7).

Листинг 4.7. Функция «ОбработатьФоновымЗаданием»

```
Функция ОбработатьФоновымЗаданием(КлючСлужебногоОбсуждения) Экспорт  
  
    ИдентификаторЗадания = Строка(Новый УникальныйИдентификатор);  
    ПараметрыЗадания = Новый Массив;  
    ПараметрыЗадания.Добавить(КлючСлужебногоОбсуждения);  
    ПараметрыЗадания.Добавить(ИдентификаторЗадания);
```

```
ФоновыеЗадания.Выполнить(  
    "СлужебныеСообщения.ВыполнитьДлительнуюОперациюНаСервере", ПараметрыЗадания);
```

```
Возврат ИдентификаторЗадания;
```

```
КонецФункции
```

В функции мы сначала создаем новый строковый уникальный идентификатор и запоминаем его, а также ключ служебного сообщения в массиве `ПараметрыЗадания`. Затем инициируем выполнение фонового задания на сервере с помощью метода `Выполнить()` менеджера фоновых заданий. В первом параметре метода мы передаем полное имя процедуры серверного модуля, которую нужно выполнить, а во втором параметре передаем массив параметров задания, созданный нами ранее.

В заключение возвращаем идентификатор фонового задания, присвоенный ему перед запуском.

Процедуру `ВыполнитьДлительнуюОперациюНаСервере()` мы также поместим в общем серверном модуле `СлужебныеСообщения` и заполним ее следующим образом (листинг 4.8).

Листинг 4.8. Процедура «ВыполнитьДлительнуюОперациюНаСервере»

```
Процедура ВыполнитьДлительнуюОперациюНаСервере(  
    КлючСлужебногоОбсуждения, ИдентификаторЗадания) Экспорт  
  
    СлужебноеОбсуждение =  
        СистемаВзаимодействия.ПолучитьОбсуждение(КлючСлужебногоОбсуждения);  
  
    СтруктураДанных = Новый Структура;  
    СтруктураДанных.Вставить("ИдентификаторЗадания", ИдентификаторЗадания);  
    СтруктураДанных.Вставить("Значение", 0);  
    СтруктураДанных.Вставить("СпособОбработки", "Индикатор");  
  
    Для Счетчик = 1 По 100 Цикл  
  
        СообщениеКлиенту =  
            СистемаВзаимодействия.СоздатьСообщение(СлужебноеОбсуждение.Идентификатор);  
        СтруктураДанных.Значение = Счетчик;  
        СообщениеКлиенту.Данные = СтруктураДанных;  
        СообщениеКлиенту.Записать();  
  
    КонецЦикла;
```

```
СообщениеКлиенту =  
    СистемаВзаимодействия.СоздатьСообщение(СлужебноеОбсуждение.Идентификатор);  
СтруктураДанных.СпособОбработки = "НеОтслеживать";  
СообщениеКлиенту.Данные = СтруктураДанных;  
СообщениеКлиенту.Записать();
```

КонецПроцедуры

В этой процедуре мы не будем реализовывать реальную длительную операцию. Покажем только схему отправки сообщений с сервера с информацией о выполнении того или иного этапа фонового задания.

Сначала по ключу служебного обсуждения, переданному в процедуру в первом параметре, мы получаем служебное обсуждение, в которое нужно отправлять сообщения о ходе выполнения процедуры.

Затем создаем структуру данных с полями ИдентификаторЗадания, Значение и СпособОбработки. Полю структуры ИдентификаторЗадания присваиваем идентификатор задания, переданный в процедуру во втором параметре. Полю СпособОбработки присваиваем строку «Индикатор», которая будет сигнализировать клиенту о том, что надо передвинуть указатель в панели состояния.

Затем организуем цикл до ста. 100 – это условное количество этапов выполнения фонового задания. На каждом этапе мы формируем сообщение, которое посылается в служебное обсуждение. В данных сообщения содержится структура СтруктураДанных. Поле Значение этой структуры отражает текущую итерацию цикла.

После выхода из цикла мы посылаем в служебное обсуждение заключительное сообщение со значением поля структуры СпособОбработки – «НеОтображать», которое сигнализирует клиенту о завершении фонового задания.

Теперь пришло время рассмотреть, как осуществляется обработка полученных сообщений на клиенте, подписанном на служебное обсуждение.

Эта обработка выполняется в процедуре `ОбработкаСообщенийСервера()` общего модуля `СлужебныеСообщенияКлиент` (листинг 4.9). Первым параметром в обработчик оповещения передается копия объекта `СообщениеСистемыВзаимодействия`, то есть того сообщения, которое было получено в служебном обсуждении.

Листинг 4.9. Процедура «ОбработкаСообщенийСервера»

```
Процедура ОбработкаСообщенийСервера(Сообщение, ДопПараметры) Экспорт
```

```
    ИндексЗадания =  
        глИдентификаторыФоновыхЗаданий.Найти(Сообщение.Данные.ИдентификаторЗадания);  
    Если НЕ ИндексЗадания = Неопределено Тогда
```

```
        Если Сообщение.Данные.СпособОбработки = "Индикатор" Тогда  
            Состояние("Выполняется обработка данных", Сообщение.Данные.Значение);  
        ИначеЕсли Сообщение.Данные.СпособОбработки = "НеОтслеживать" Тогда  
            глИдентификаторыФоновыхЗаданий.Удалить(ИндексЗадания);  
        КонецЕсли;
```

```
    КонецЕсли;
```

```
КонецПроцедуры
```

Сначала идентификатор задания, переданный в данных сообщения (`Сообщение.Данные.ИдентификаторЗадания`), мы ищем в глобальном массиве `глИдентификаторыФоновыхЗаданий`, в котором мы его запомнили перед запуском фонового задания. Если идентификатор найден, значит, это то фоновое задание, которое было запущено на этом клиенте.

В этом случае, в зависимости от способа обработки (`Сообщение.Данные.СпособОбработки`), мы либо передвигаем индикатор в панели состояния, либо удаляем идентификатор из массива отслеживаемых заданий, потому что задание завершилось.

Таким образом, сначала кадровик с помощью команды Создание служебного обсуждения создает служебное обсуждение, в котором он будет получать сообщения от сервера. Затем подписывается на это обсуждение, выполнив команду Подписка на служебное обсуждение. После этого, запустив регламентированный отчет с помощью команды Запустить длительную операцию, он увидит панель состояния, показывающую ход выполнения длительной операции на сервере, и сможет визуально оценить, когда примерно она завершится (рис. 4.1).

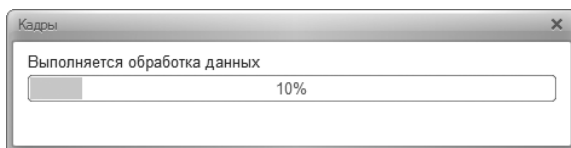


Рис. 4.1. Панель состояния

Отправка уведомлений на клиент при наступлении событий на сервере

В этом разделе мы рассмотрим случай, когда при наступлении каких-то событий на сервере сервер через служебное обсуждение информирует об этом клиентское приложение. И оно, в свою очередь, реагирует на это сообщение.

Например, в тот момент, когда менеджер интернет-магазина формирует и проводит заказ со статусом В работе, курьеру, указанному в заказе, должно автоматически прийти уведомление об этом. После этого на клиенте у соответствующего курьера должна быть автоматически открыта форма проведенного заказа.

В альтернативном варианте эту функциональность можно реализовать как взаимодействие менеджера и курьера или как

взаимодействие бота и курьера, которое мы рассматривали в предыдущей главе. Но в данном случае нас интересует именно аспект программного взаимодействия сервера и клиентского приложения.

Чтобы реализовать нашу задачу, нужно создать неотображаемое служебное обсуждение, участниками которого будут не все сотрудники интернет-магазина, а только курьеры, то есть пользователи с соответствующей ролью. Затем каждый раз при запуске клиентского приложения оно должно подписываться на это служебное обсуждение. При этом к обсуждению будет подключаться обработчик, который вызывается после появления новых сообщений в обсуждении. В этом обработчике после получения сообщения о проведении заказа у курьера, которому предназначается этот заказ, будет появляться уведомление и открываться форма заказа.

ПРИМЕЧАНИЕ

Пример отправки уведомлений курьерам с сервера можно посмотреть в конфигурации «Программное взаимодействие Напоминание Курьерам», которая прилагается к книге на компакт-диске.

Итак, сначала создадим служебное обсуждение для получения курьерами информации с сервера. Сделаем это в момент программного создания пользователей системы взаимодействия. Для упрощения примера будем реализовывать соответствующий код с помощью общей команды НачатьРаботуССистемойВзаимодействия, которую нужно вызвать всего один раз (листинг 4.10).

Листинг 4.10. Процедура «ОбработкаКоманды»

```
&НаКлиенте  
Процедура ОбработкаКоманды(ПараметрКоманды, ПараметрыВыполненияКоманды)  
  
    СлужебныеСообщения.Инициализация();  
    СлужебныеСообщенияКлиент.ПриНачалеРаботы();  
  
КонецПроцедуры
```

В обработчике команды мы вызываем процедуру Инициализация() общего серверного модуля СлужбныеСообщения (листинг 4.11).

Листинг 4.11. Процедура «Инициализация»

Процедура Инициализация() Экспорт

```
Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда  
    Возврат;  
КонецЕсли;
```

```
УстановитьПривилегированныйРежим(Истина);
```

```
КлючСлужебногоОбсуждения = СлужбныеСообщения.ПолучитьКлючСлужебногоОбсуждения();
```

```
// Создаем пользователей системы взаимодействия
```

```
Для Каждого ПользовательИБ ИЗ ПользователиИнформационнойБазы.ПолучитьПользователей() Цикл
```

```
    Попытка
```

```
        ИдентификаторПользователяСВ =
```

```
            СистемаВзаимодействия.ПолучитьИдентификаторПользователя(  
                ПользовательИБ.УникальныйИдентификатор);
```

```
        ПользовательСВ =
```

```
            СистемаВзаимодействия.ПолучитьПользователя(ИдентификаторПользователяСВ);
```

```
    Исключение
```

```
        ПользовательСВ = СистемаВзаимодействия.СоздатьПользователя(ПользовательИБ;  
            ПользовательСВ.Записать());
```

```
    КонецПопытки;
```

```
Если ПользовательИБ.Роли.Содержит(Метаданные.Роли.Курьер) Тогда
```

```
    // создаем обсуждение служебное обсуждение для курьеров
```

```
    СлужебноеОбсуждение = СистемаВзаимодействия.ПолучитьОбсуждение(  
        КлючСлужебногоОбсуждения);
```

```
    Если СлужебноеОбсуждение = Неопределено Тогда
```

```
        СлужебноеОбсуждение = СистемаВзаимодействия.СоздатьОбсуждение();
```

```
        СлужебноеОбсуждение.Отображаемое = Ложь;
```

```
        СлужебноеОбсуждение.Ключ = КлючСлужебногоОбсуждения;
```

```
    КонецЕсли;
```

```
    Если НЕ СлужебноеОбсуждение.Участники.Содержит(  
        ПользовательСВ.Идентификатор) Тогда
```

```
        СлужебноеОбсуждение.Участники.Добавить(ПользовательСВ.Идентификатор);
```

```
    КонецЕсли;
```

```
    СлужебноеОбсуждение.Записать());
```

```
КонецЕсли;
```

```
КонецЦикла;
```

```
КонецПроцедуры
```

В этой процедуре сначала создаются пользователи системы взаимодействия на основе пользователей информационной базы. Этот вопрос подробно рассматривался в предыдущей главе, поэтому мы не будем еще раз на нем останавливаться. Нужный нам фрагмент выделен в листинге процедуры жирным шрифтом.

Ключ служебного обсуждения получаем с помощью функции `ПолучитьКлючСлужебногоОбсуждения()` общего модуля `СлужебныеСообщения` как произвольную строку (листинг 4.12).

В процессе создания пользователей системы взаимодействия для пользователей с ролью Курьер мы создаем служебное обсуждение с ключом `КлючСлужебногоОбсуждения`, если обсуждение с таким ключом еще не существует. И затем добавляем созданного пользователя системы взаимодействия в состав участников этого обсуждения. Чтобы обсуждение не отображалось в интерфейсе, мы устанавливаем свойство обсуждения `Отображаемое` в значение `Ложь`.

Листинг 4.12. Функция «ПолучитьКлючСлужебногоОбсуждения»

```
Функция ПолучитьКлючСлужебногоОбсуждения() Экспорт
    Возврат "Служебное обсуждение для курьеров";
КонiecФункции
```

Теперь из модуля приложения в обработчике события `ПриНачалеРаботыСистемы` вызовем процедуру `ПриНачалеРаботы()` общего клиентского модуля `СлужебныеСообщенияКлиент` (листинг 4.13).

Листинг 4.13. Обработчик события «ПриНачалеРаботыСистемы» в модуле приложения

```
Процедура ПриНачалеРаботыСистемы()
    СлужебныеСообщенияКлиент.ПриНачалеРаботы();
КонiecПроцедуры
```

В результате при каждом запуске клиентского приложения оно будет подписываться на получение сообщений в служебном обсуждении. Кроме того, эта же процедура вызывается и в обработчике команды НачатьРаботуССистемойВзаимодействия после создания служебного обсуждения (см. листинг 4.10).

Процедуру ПриНачалеРаботы() общего модуля СлужебныеСообщенияКлиент заполним следующим образом (листинг 4.14).

Листинг 4.14. Процедура «ПриНачалеРаботы»

Процедура ПриНачалеРаботы() Экспорт

```
Если НЕ СистемаВзаимодействия.ИнформационнаяБазаЗарегистрирована() Тогда  
    Возврат;  
КонецЕсли;
```

```
Если НЕ СлужебныеСообщения.ЭтаПодпискаДляКурьера() Тогда  
    Возврат;  
КонецЕсли;
```

```
КлючСлужебногоОбсуждения = СлужебныеСообщения.ПолучитьКлючСлужебногоОбсуждения();  
ОбработкаСообщенийСервера = Новый ОписаниеОповещения(  
    "ОбработкаСообщенийСервера", СлужебныеСообщенияКлиент);  
СистемаВзаимодействия.НачатьПодключениеОбработчикаНовыхСообщений(  
    КлючСлужебногоОбсуждения, ОбработкаСообщенийСервера);
```

КонецПроцедуры

В этой процедуре мы сначала должны определить, «кем» по своей роли является текущий пользователь клиентского приложения. Потому что никто, кроме курьеров, не является участником служебного обсуждения и не должен на него подписываться, иначе будет получена ошибка. Эту проверку мы выполняем с помощью функции ЭтаПодпискаДляКурьера() общего модуля СлужебныеСообщения (листинг 4.15).

Затем мы получаем ключ служебного обсуждения и вызываем метод менеджера системы взаимодействия НачатьПодключениеОбработчикаНовыхСообщений(). Во втором параметре метода мы передаем ключ служебного обсуждения. В третьем параметре мы передаем описание обработчика оповещения – процедуры

ОбработкаСообщенийСервера(), которая будет вызвана после появления новых сообщений в обсуждении с ключом КлючСлужебногоОбсуждения.

Таким образом, при получении новых сообщений в нашем служебном обсуждении будет выполняться процедура ОбработкаСообщенийСервера(), в которой и производится обработка данных, содержащихся в сообщениях. Сам обработчик мы рассмотрим позже, после того как рассмотрим процедуру обработки проведения документов Заказ, в которой эти сообщения отправляются с сервера (см. листинг 4.16).

Листинг 4.15. Функция «ЭтаПодпискаДляКурьера»

Функция ЭтаПодпискаДляКурьера() Экспорт

```
ПользовательИБ = ПользователиИнформационнойБазы.ТекущийПользователь();  
Если ПользовательИБ.Роли.Содержит(Метаданные.Роли.Курьер) Тогда  
    Возврат Истина;  
Иначе  
    Возврат Ложь;  
КонецЕсли;
```

КонецФункции

Теперь в процедуру ОбработкаПроведения() в модуле документа Заказ нам нужно добавить код для отправки уведомлений курьерам в служебное обсуждение, в случае если заказ имеет статус ВРаботе (листинг 4.16).

Листинг 4.16. Процедура «ОбработкаПроведения» в модуле документа «Заказ»

Процедура ОбработкаПроведения(Отказ, Режим)

```
Если СтатусЗаказа = Перечисления.СтатусыЗаказов.ВРаботе Тогда
```

```
    // регистр ОстаткиТоваров Приход  
    Движение.ОстаткиТоваров.Записывать = Истина;  
    Для Каждого ТекСтрокаТовары Из Товары Цикл  
        Движение = Движения.ОстаткиТоваров.Добавить();  
        Движение.ВидДвижения = ВидДвиженияНакопления.Приход;  
        Движение.Период = Дата;  
        Движение.Товар = ТекСтрокаТовары.Товар;  
        Движение.Цвет = ТекСтрокаТовары.Цвет;
```

```
Движение.Размер = ТекСтрокаТовары.Размер;  
Движение.Склад = Склад;  
Движение.Заказано = ТекСтрокаТовары.Количество;  
КонецЦикла;  
  
КлючСлужебногоОбсуждения =  
    СлужебныеСообщения.ПолучитьКлючСлужебногоОбсуждения();  
СлужебныеСообщения.ОтправкаУведомленийКурьерам(  
    КлючСлужебногоОбсуждения, Ссылка, Курьер);  
  
ИначеЕсли СтатусЗаказа = Перечисления.СтатусыЗаказов.Выполнен Тогда  
  
    ...  
  
КонецЕсли;  
  
    ...  
  
КонецПроцедуры
```

В выделенном фрагменте процедуры мы получаем ключ служебного обсуждения. И затем вызываем процедуру `ОтправкаУведомленийКурьерам()` общего модуля `СлужебныеСообщения` и передаем в нее этот ключ, ссылку на заказ и ссылку на курьера, указанного в заказе (листинг 4.17).

Листинг 4.17. Процедура «ОтправкаУведомленийКурьерам»

```
Процедура ОтправкаУведомленийКурьерам(КлючСлужебногоОбсуждения, Заказ, Курьер) Экспорт  
  
    СлужебноеОбсуждение =  
        СистемаВзаимодействия.ПолучитьОбсуждение(КлючСлужебногоОбсуждения);  
    СтруктураДанных = Новый Структура();  
    СообщениеКлиенту =  
        СистемаВзаимодействия.СоздатьСообщение(СлужебноеОбсуждение.Идентификатор);  
  
    СтруктураДанных.Вставить("Текст", "Сформирован новый заказ");  
    СтруктураДанных.Вставить("Заказ", Заказ);  
    СтруктураДанных.Вставить("Курьер", Курьер);  
    СообщениеКлиенту.Данные = СтруктураДанных;  
    СообщениеКлиенту.Записать();  
  
КонецПроцедуры
```

В этой процедуре по ключу служебного обсуждения, переданному в процедуру в первом параметре, мы получаем служебное обсуждение, в которое нужно отправлять сообщения для курьеров. Затем создаем сообщение в этом обсуждении.

После этого создаем структуру данных с полями Текст, Заказ и Курьер. Полям структуры Заказ и Курьер присваиваем ссылку на проводимый заказ и ссылку на курьера, переданные в процедуру во втором и третьем параметрах. И помещаем структуру СтруктураДанных в данные сообщения.

Теперь пришло время рассмотреть, как осуществляется обработка полученных сообщений на клиенте, подписанном на служебное обсуждение.

Эта обработка выполняется в процедуре ОбработкаСообщенийСервера() общего модуля СлужебныеСообщенияКлиент (листинг 4.18). Первым параметром в обработчик оповещения передается копия объекта СообщениеСистемыВзаимодействия, то есть того сообщения, которое было получено в служебном обсуждении.

Листинг 4.18. Процедура «ОбработкаСообщенийСервера»

Процедура ОбработкаСообщенийСервера(Сообщение, ДопПараметры) Экспорт

```
ИдентификаторПользователяСВКурьер =  
    СлужебныеСообщения.ПолучитьИдентификаторСВКурьера(Сообщение.Данные.Курьер);
```

```
Если СистемаВзаимодействия.ИдентификаторТекущегоПользователя() =  
    ИдентификаторПользователяСВКурьер Тогда  
    ПоказатьЗначение(, Сообщение.Данные.Заказ);  
    ПоказатьОповещениеПользователя(Сообщение.Данные.Текст,,,  
        СтатусОповещенияПользователя.Важное);
```

```
КонецЕсли;
```

```
КонецПроцедуры
```

Сначала с помощью функции ПолучитьИдентификаторСВКурьера() общего модуля СлужебныеСообщения (см. листинг 4.19) по переданной ссылке на курьера (Сообщение.Данные.Курьер)

мы получаем идентификатор курьера в системе взаимодействия. И если он совпадает с идентификатором текущего пользователя, то, значит, это клиентское приложение курьера, указанного в заказе, при проведении которого было послано сообщение в служебное обсуждение.

В этом случае мы открываем форму заказа (Сообщение.Данные.Заказ) и показываем оповещение пользователя с текстом, переданным в сообщении (Сообщение.Данные.Текст).

Листинг 4.19. Функция «ПолучитьИдентификаторСВКурьера»

Функция ПолучитьИдентификаторСВКурьера(Курьер) Экспорт

```
ПользовательИБКурьер = ПользователиИнформационнойБазы.НайтиПоИмени(СокрЛП(Курьер.Код));
Если ПользовательИБКурьер = Неопределено Тогда
    Возврат "";
КонецЕсли;

ИдентификаторПользователяСВКурьер =
    СистемаВзаимодействия.ПолучитьИдентификаторПользователя(
        ПользовательИБКурьер.УникальныйИдентификатор);
Возврат ИдентификаторПользователяСВКурьер;
```

КонецФункции

Таким образом, сначала администратор с помощью команды Начать работу с системой взаимодействия создает служебное обсуждение, в котором курьеры будут получать сообщения от сервера. Затем каждый раз при запуске клиентские приложения всех курьеров подписываются на это обсуждение. И когда менеджер формирует и проводит заказ со статусом В работе, курьеру, указанному в заказе, приходит уведомление об этом и форма самого заказа автоматически открывается (рис. 4.2).

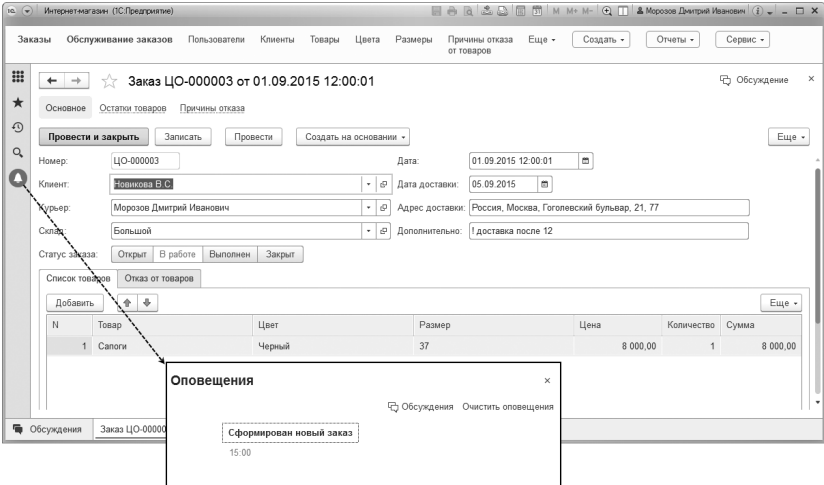


Рис. 4.2. Клиентское приложение курьера

© ООО «1С-Публишинг», 2019

© Оформление. ООО «1С-Публишинг», 2019

Все права защищены.

Материалы предназначены для личного индивидуального использования приобретателем.

Запрещено тиражирование, распространение материалов, предоставление доступа по сети к материалам без письменного разрешения правообладателей.

Разрешено копирование фрагментов программного кода для использования в разрабатываемых прикладных решениях.

Фирма «1С»

123056, Москва, а/я 64, Селезневская ул., 21

Тел.: (495) 737-92-57

1c@1c.ru, <http://www.1c.ru/>

Издательство ООО «1С-Публишинг»

127434, Москва, Дмитровское ш., 9

Тел.: (495) 681-02-21

publishing@1c.ru, <http://books.1c.ru>

Об опечатках просьба сообщать по адресу books.v8@1c.ru.