Компания “Консид Решения”

Методология разработки приложений для терминалов сбора данных на базе Open Source HH.Mobile

Оглавление

Введение

Open Handheld Engine (OHE) - платформа бизнес-приложений

Платформа решений Open HandHeld Engine является ядром линейки всех решений предлагаемых компанией. Реализация платформы обеспечивает высокую производительность и надежность выполняемых операций за счет используемых технологий и глубокой проработки внутренней архитектуры.

Платформа является основой для разработки промышленных бизнес-приложений для автоматизации при помощи конструкторов Архитектуры.

Преимущества:

• высокая производительность

• масштабируемость

• высокая степень адаптируемости

• независимость от поставщика оборудования ТСД (мобильный терминал сбора данных)

• короткие сроки разработки конечных приложений за счет использования визуальных средств разработки

Платформа бизнес-приложений включает в себя:

• WEB-сервер

• Telnet-сервер

Web-сервер обеспечивает работу тонких клиентов на стационарных компьютерах. Web-сервер основан на технологии asp.net, но использует свои управляющие элементы, обеспечивающие большую производительность за счет минимизации времени по учету состояний объектов и использования оптимизированной логики работы. Web-сервер позволяет писать клиентские приложения для доступа и манипулирования данными, хранящимися в БД.

Пользователю нет необходимости обладать знаниями в области разработки web-приложений. Всю внутреннюю обработку сервер берет на себя за счет использования конфигурации.

Конфигурация представляет собой набор xml-файлов, которые компилируются в приложении при запуске системы.

Для разработки конфигурации конечного приложения администратор системы должен обладать только навыками разработки запросов к базам данных и хранимых процедур, и

пройти курс обучения технологии разработки WEB-приложений при помощи конструктора Open Source HH.WEB.

Наличие конфигурации сервера и использование конструктора web-приложений обеспечивает следующие преимущества:

• ускоряет разработку бизнес-приложений в разы

• минимизирует набор специфических знаний разработчика

• обеспечивает стабильную работу web-приложений

• сокращение времени на постановку технического задания. Данный подход за счет унификации стандартов разработки позволяет сократить время на постановку технического задания, так как в этом случае будет описываться реализация бизнес-логики, а в постановке задачи технической реализации не будет необходимости.

• сокращение времени тестирования нового функционала. Данный подход за счет унификации стандартов разработки позволяет сократить время тестирования разработанной функциональности конечного приложения, так как, в этом случае, будет тестироваться только реализация бизнес-логики, а тестирование программной реализации отсутствует, в связи с тем, что вся программная логика находится в ядре системы и она уже оттестирована.

Telnet-сервер представляет собой набор стандартных обработчиков, обеспечивающих обмен данными с радиотерминалами.

Telnet-сервер платформы Open HandHeld Engine обеспечивает работу по протоколу telnet с использованием любых клиентов, поддерживающих данный протокол.

Логика работы Telnet-сервера вынесена в конфигурационный файл формата xml. Это обеспечивает широкие возможности адаптации работы Telnet-сервера без трудоемкой разработки.

Для разработки терминальных приложений под Telnet-сервер используется визуальный конфигуратор Open Source HH.Mobile. Использование конфигуратора ускоряет процесс разработки приложений и не требует специфических навыков персонала. Это сокращает стоимость дополнительных доработок, которые могут потребоваться в будущем. Данный подход также позволяет сократить время на тестирование терминального приложения.

Высокая производительность Telnet-сервера обеспечивается наличием изолированных потоков для каждого тонкого клиента, что исключает возможность “зависания” Telnet-сервера при выполнении ресурсоемких операций пользователями.

В данном документе рассматривается методология разработки приложений для терминалов сбора данных при помощи визуального средства разработки Open Source HH.Mobile.

1 Цели и задачи, решаемые на этапах разработки бизнес-приложений для ТСД

Open HandHeld.Engine (OHE) в своем составе имеет визуальное средство для разработки конфигурации приложений для терминалов сбора данных Open Source HandHeld.Mobile, предназначенное для облегчения и ускорение процесса разработки бизнес-процессов приложения.

Бизнес-процессы приложения для ТСД рекомендуется разрабатывать по следующей методике. Это обеспечит быструю и качественную разработку решений.

В разработке решений рекомендуется участие двух человек (возможно совмещение ролей в одном человеке):

- бизнес-аналитика
- разработчика SQL

Задачами бизнес-аналитика являются:

- Согласование проектной документации, сроков разработки бизнес процессов
- Определение основных объектов, используемых в разработке и их связь с другими бизнес процессами
- Разработка и Согласование диаграмм (блок-схем) бизнес-процессов с Заказчиком
- Постановка задач на разработку хранимых процедур для программиста
- Визуальная разработка приложения под OHE

Задачами SQL разработчика являются:

- Разработка хранимых процедур в соответствие с требованиями бизнес-аналитика
- Тестирование бизнес-процессов
- Внутренняя приемка-сдача разработанных процессов

Возможно совмещение обязанностей бизнес-аналитика и программиста в одном лице, но не рекомендуется.

2 Этапы разработки бизнес-приложений для ТСД

Разработка решений на платформе ОНЕ включает следующие этапы:

1. Разработка и Согласование диаграмм разрабатываемых бизнес-процессов.

2. Эскизное проектирование. Разработка экранов терминалов с описанием свойств и событий при помощи конструктора архитектуры Open Source HH.Mobile (Визуальное проектирование).

3. Разработка хранимых процедур (SQL процедуры для выборки данных и проводки изменений в БД).

4. Тестирование разработанного бизнес-процесса.

5. Сдача в эксплуатацию.

2.1 Согласование диаграмм разрабатываемых бизнес-процессов

Этот этап методологии разработки бизнес-процессов предполагает плотное общение бизнес-аналитика со стороны компании интегратора (или ИТ-отдела) с заказчиком - бизнес-аналитиком или логистом от производства. На данном этапе определяются и описываются бизнес-процессы для дальнейшей их реализации в Open Source HH.Mobile.

Результатом этого этапа являются согласованные диаграммы бизнес-процессов, которые подписываются обеими сторонами, и представляют собой техническое задание на разработку бизнес-процессов.

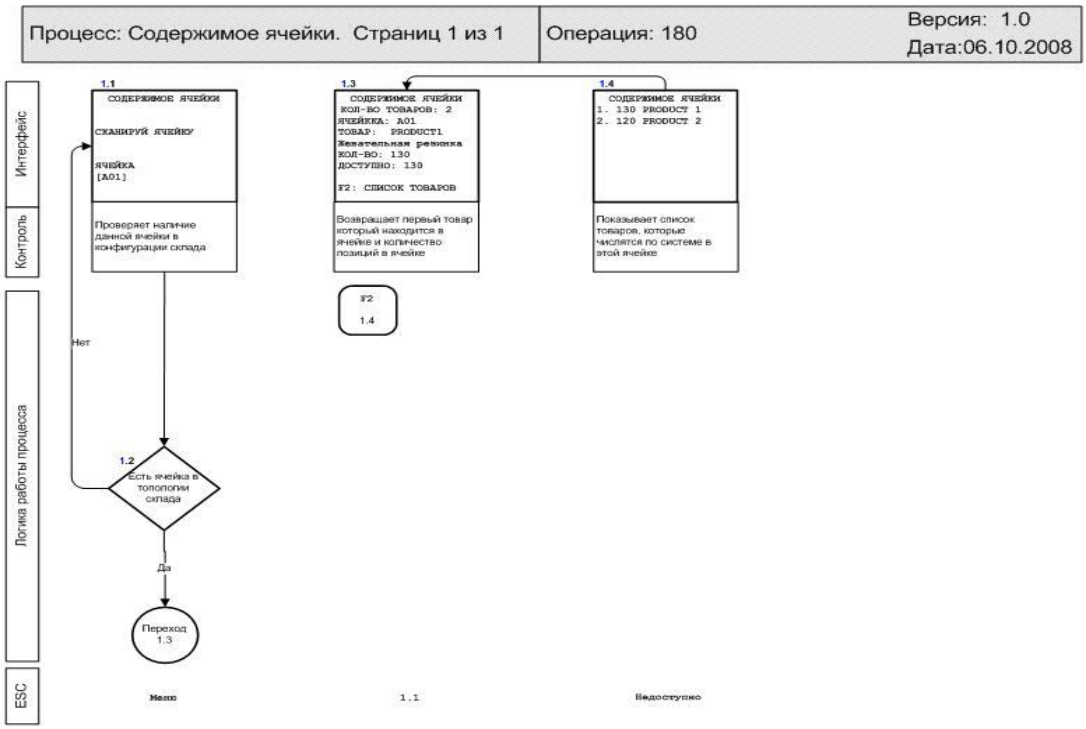


Рис. 2.1. Пример диаграммы бизнес-процесса

Результатом этапа являются диаграммы бизнес-процессов, которые описывают все экраны терминалов и логику работы бизнес-процесса.

Рекомендуется разрабатывать диаграммы в формате Ms Visio. Но формат представления диаграммы бизнес-процесса может быть различен. Главное, чтобы разработчик приложения мог получить всю необходимую информацию.

2.2 Эскизное (Визуальное) проектирование.

На этом этапе все экраны терминалов и логика бизнес процессов в соответствии с согласованными блок-схемами переносятся в конфигурацию приложения Open Source HH.Mobile. Таким образом определяется последовательность экранов и логика работы приложения.

Средство разработки Open Source HH.Mobile имеет встроенное отображение бизнес-процессов в виде диаграмм, которое может использоваться для соответствия подписанных и разрабатываемых бизнес-процессов.

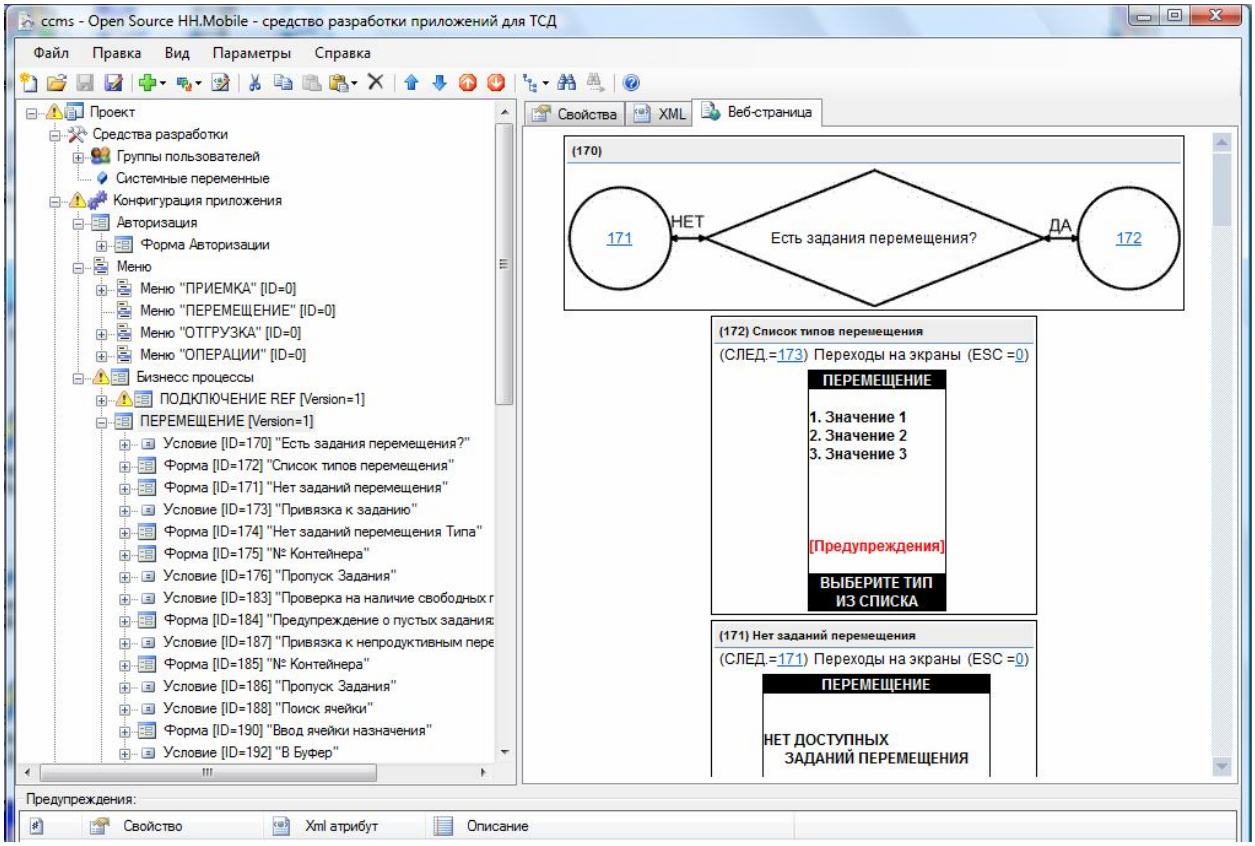


Рис. 2.2. Визуализация конфигурации приложения и разработанного бизнес-процесса

На этапе эскизного проектирования выполняются следующие работы:

- создается отображение всех экранных форм бизнес-процесса
- определяются логические переходы между экранами

- определяются основные элементы общения пользователя. Т.е. данные, которые должен вводить пользователь во время работы (сканировать ячейку, товар, коробку и т.п.)
- определяются основные события (инициализации, проверки и т.п.), которые будут использоваться в бизнес-процессе
- определяются имена хранимых процедур и параметров, которые будут использоваться.
- программистом создаются пустые хранимые процедуры, возвращающие константы. Используется для тестов логики работы бизнес-процесса.
- тестируется логика работы приложения в соответствие с ТЗ. Исправляются ошибки и недочеты.

Результатом окончания этапа является – конфигурация приложения с полным описанием экранов приложения, логики работы бизнес-процессов и вызовом хранимых процедур для обработки и связи с базой данных приложения.

Для ускорения разработки бизнес-процессов методология может быть изменена – изменен порядок выполнения этапов, первый и второй этап меняются местами или совмещаются. Тогда в соответствии с измененной методологией последовательность будет следующая: сначала бизнес-аналитик согласует словесное описание логики бизнес-процессов; затем процесс реализуется в Open Source HH.Mobile и распечатывается его диаграмма из соответствующего экрана приложения (данная диаграмма используется в качестве черновика); далее по черновикам диаграмм процессов проверяется правильность реализации, по итогам проверки в процесс вносятся изменения; далее бизнес-аналитик в соответствии с черновиками бизнес-процесса переносит блок-схему процесса в Visio, распечатывает и согласовывает с заказчиком. После согласования разрабатываемых процессов производится разработка хранимых процедур, тестирование, отладка а затем сдача в эксплуатацию.

2.3 Разработка хранимых процедур

Список хранимых процедур составляется бизнес-аналитиком совместно с разработчиком на этапе согласования бизнес-процессов и на этапе эскизного проектирования.

Хранимые процедуры разрабатываются в соответствии с требованиями и ограничениями на основе шаблонов. Каждому шаблону хранимой процедуры соответствует свой набор входных параметров.

2.4 Тестирование

После эскизного проектирования и разработки хранимых процедур можно приступить к тестированию приложения. Для этого необходимо установить сервер приложения сервер базы данных, сконфигурировать сервер приложения, перенести конфигурацию приложения. Тестирование бизнес-процессов можно выполнять на эмуляторе терминалов сбора данных, например KoalaTerm или встроенном эмуляторе телнет операционной системы Windows. Данные для тестирования инициируются с диспетчерского WEB приложения. Результаты выполнения процессов просматриваются с диспетчерского WEB приложения. Для отслеживания ошибок в хранимых процедурах и времени их выполнения в SQL Manager запускается приложение Prifiler.

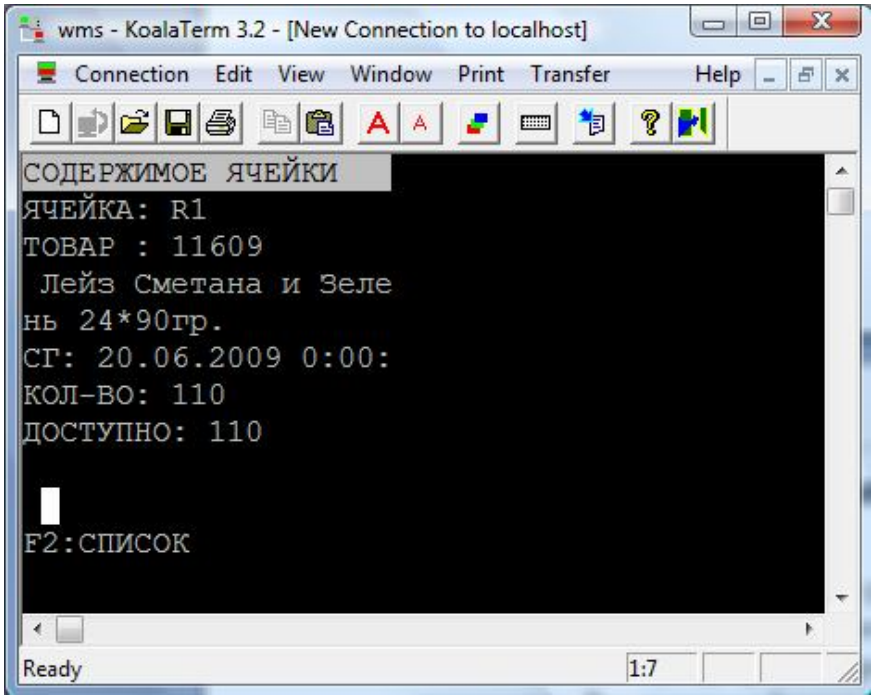


Рис. 2.3. Эмуляция работы бизнес-процесса из приложения KoalaTerm

Если в ходе тестирования обнаружены ошибки, то в этом случае анализируются логи приложения и MsSql Profiler, база данных приложения и конфигурация приложения. Найденные ошибки устраняются и далее, продолжается тестирование.

По этогам тестирования подписывается акт приемки сдачи результатов тестирования.

3 Архитектура и общее описание

3.1 Используемые термины и сокращения

Телнет Сервер или Сервер - Программный Комплекс, обеспечивающий взаимодействие между пользователями системы, работающими с Терминалами и Сервером, содержащим ядро для разработки бизнес-процессов клиента.

Терминал сбора данных или Терминал - Прибор, представляющий собой микрокомпьютер, оснащенный сканером штрихкода, процессором, операционной системой, программой, встроенной памятью и системой ввода-вывода информации (экран и клавиатура) в виде, понятном человеку, а также системой обмена данными с компьютером.

Сканер штрихкода или Сканер - прибор, входящий в состав Терминала или выполненный в виде отдельного прибора. Предназначен для считывания и распознавания (декодирования) штриховых кодов и передачи результатов декодирования в терминал сбора данных или базу данных предприятия.

Принтер Штрихкода или Принтер - прибор, предназначенный для печати Этикеток, и используемый для нанесения на Этикетку необходимых данных.

Оператор - человек, обученный работе с Комплексом и обладающий навыками работы с оборудованием и Базой.

АРМ - Автоматизированное Рабочее Место оператора

Сервер БД – Компьютер, предназначенный для хранения и обработки данных, поступающих от Модулей и АРМ.

Сервер – часть Программы, находящаяся на персональном компьютере и необходимая для обмена данными между компьютером и Терминалом.

Задание для принтера Набор данных для печати этикеток на определенном принтере.

Открытие задания Операция запуска заданий для принтеров.

Закрытие задания Операция, завершающая печать этикеток на принтере

Событие инициализации – событие, возникающее до появления экранной формы или управляющего элемента на экране терминала.

Событие перехода – событие, возникающее после обработки всех управляющих элементов на экранной форме.

Событие обновления – событие, возникающее после обработки управляющего элемента на экранной форме.

3.2 Архитектура и общее описание системы

Приложение для управления радио-терминалами сбора данных представляет собой windows-службу, обеспечивающий работу с радио-терминалами по протоколу telnet. Сервер и Терминалы общаются по радиосети в стандарта Wi-Fi.

Терминалы должны поддерживать протокол VT220.

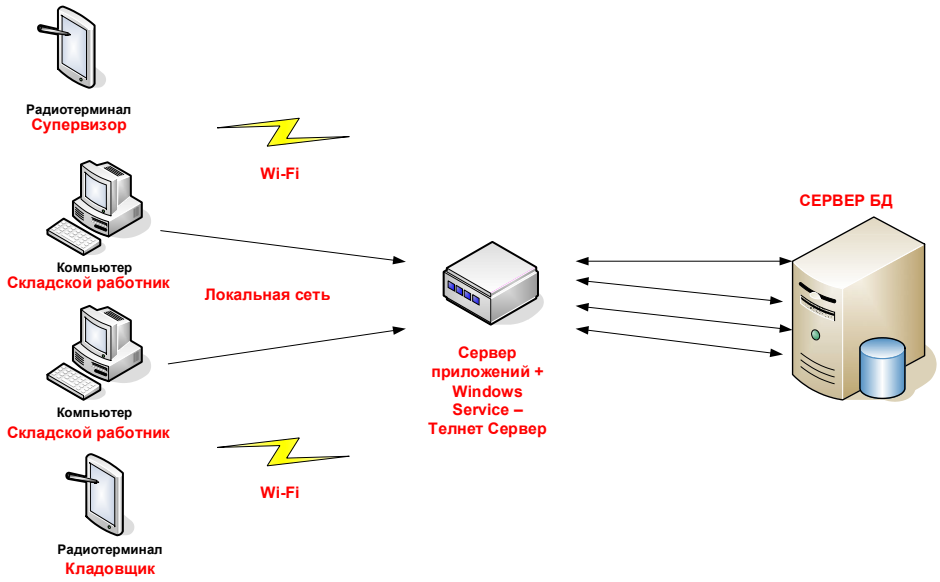


Рис. 3.1. Архитектура Телнет Сервера.

Сервер реализован как часть трехуровневой архитектуры и представляет собой сервер приложений, в котором вся логика работы и экранные формы радиотерминалов вынесены в конфигурацию. Телнет сервер не имеет оконных форм и реализован как служба Windows.

Сервер представляет собой многопоточное приложение, в котором каждый клиент (Терминал) работает в изолированном потоке. Это позволяет избежать зависания Сервера, если один из Операторов выполняет ресурсоемкую операцию.

Сервер не привязан к Базе Данных. Работа с Базой Данных поддерживается посредством вызова хранимых процедур. В качестве баз данных могут быть использованы MSSQL2000/2005, Oracle и т.п. Администратору, работающему с Сервером необходимо просто описать логику работы с данными в хранимых процедурах и описать их вызов в конфигурации приложения.

Сервер обеспечивает работу на любом порту, который настраивается в конфигурации приложения.

Большинство настроек сервера вынесены в файлы конфигурации, и считываются при запуске Сервера. Конфигурация представляет собой xml файлы интуитивно понятного формата. Разработка конфигурации сервера приложения осуществляется в визуальном приложении Open Source HH.Mobile.

Конфигурация сервера приложений разделена на два файла:

- system.xml – содержит системные настройки Сервера. В конфигурацию вынесены, порты подключения Сервера, клавиши работы с меню Терминала, размеры экрана, строка подсоединения к Базе Данных.

- config.xml – содержит описание экранных форм терминалов, вызываемых хранимых процедур, переходов, между экранами пользователей, меню терминала, описание привилегий и пользователей Терминала.

При возникновении ошибок, связанных с работой Сервера, все исключения и ошибки отслеживаются и записываются в виде текстовых файлов в директорию Log, которая находится в инсталляционной директории приложения. Файлы логов разбиты по дате появления в них информации и представляют собой текстовые файлы.

3.3 Спецификация системы

Описание пользователей

Все пользователи, которые работают с системой, должны быть зарегистрированы в системе и включены в одну из функциональных групп, таких как администраторы, супервизоры, операторы и тд.

Описание меню Терминалов и экранных форм бизнес-процессов

В конфигурации должно быть описано функциональное меню приложения.

Каждый пункт меню приложения в конфигурации описывается узлом – **menu**. При описании меню задаются свойства пункта меню и переходы на другие пункты меню или вызов бизнес-процесса.

В конфигурации приложения должны быть описаны все бизнес-процессы приложения в виде совокупности экранных форм, состоящих из множества объектов для которых задаются свойства и события.

Описание объектов (управляющих элементов) экранных форм.

Все экранные формы состоят из списка управляющих элементов (объектов), с которыми работает пользователь в процессе выполнения бизнес-процессов.

Все управляющие элементы разделяются на шесть типов:

text - Представляет собой текст, который будет виден на экране терминала. Это статический текст, который может содержать подсказки или какие-то статические сообщения.

input - Объект ввода пользовательских данных.

Все элементы управления типа input могут поддерживать два события

- init - инициализация контрола. Автоматически вызывается при попадании пользователя на форму (если событие определено для контрола).
- change - событие обновления контрола. Атоматически вызывается при введении пользователем данных в приложении (если событие определено для контрола).

Кроме этого, ввод данных от пользователя проверяется на соответсвие типу данных элемента управления.

- list - описывает список значений, которые выбираются из этого списка.

- **function** – Управлющий элемент, который определяет наличие функциональных клавиш на экранной форме (например F1…F15)

- **help** – содержит статический текст, используется для подсказок на экранных формах, при нажатии на клавишу помощи.

- **message** – содержит данные, возвращенные хранимой процедурой, обычно описание ошибок, или исключений.

3.4 Политика безопасности и лицензирование

Файловая система

Все файлы с расширениями xml и файлы *.log должны быть доступны как для чтения, так и для изменений. Временные файлы при работе программы создаются в папке программы, поэтому она должна быть помечена как доступная для изменений.

Сервер должен запускаться от имени пользователя, для которого должен быть обеспечен доступ к папке инсталляции Сервера на чтение и запись.

При Windows аутентификации на MSSQL сервере пользователь должен иметь доступ к Базе Данных.

Лицензирование

Сервер поставляется вместе с комплектом пользовательских лицензий и лицензии программного обеспечения, кроме этого включаются 2 лицензии программного обеспечения для разработчиков.

Лицензия представляет собой xml файл, который содержит информацию о максимальном количестве одновременно работающих пользователей, доступных модулях, информацию о фирме и информацию, об уникальном ключе, сгенерированным Телнет Сервером.

Одна лицензия соответствует одному серверу приложений. При необходимости использования Телнет Сервера на нескольких серверах приложений необходимо приобретение нескольких лицензий.

Процесс получения лицензии прост:

Разработчику высылается файл, содержащий ключ и информацию о фирме, который генерируется в процессе установки программного обеспечения. Файл находится в корневом каталоге, на диске где установлена операционная система. Далее файл посылается разработчику. В ответ на присланный ключ разработчик высылает файл с лицензиями для данного сервера. Затем пользователь заменяет сгенерированный Сервером файл на файл, присланный разработчиком.

4 Функционал приложения OpenSourceHH.Mobile.

4.1 Используемые термины и сокращения

Экранная форма (форма) – это содержимое экрана, которое отражается на экране терминала.

События – реакция на действия, которые произвел пользователь во время работы, или которые возникли автоматически.

Событие инициализации – событие, возникающее до появления экранной формы или управляющего элемента на экране терминала.

Событие перехода – событие, возникающее после обработки всех управляющих элементов на экранной форме.

Событие обновления – событие, возникающее после обработки управляющего элемента на экранной форме.

Событие выхода – событие, возникающее при нажатии пользователем клавиши выхода с формы.

Управляющий элемент экранных форм(контрол) – часть информации, которая может отображаться на экране терминала. С ним могут связаны события, на основе данных введенных пользователем.

Терминал сбора данных или Терминал - Прибор, представляющий собой микрокомпьютер, оснащенный сканером штрихкода, процессором, операционной системой, программой, встроенной памятью и системой ввода-вывода информации (экран и клавиатура) в виде, понятном человеку, а также системой обмена данными с компьютером.

Сканер штрихкода или Сканер - прибор, входящий в состав Терминала или выполненный в виде отдельного прибора. Предназначен для считывания и распознавания (декодирования) штриховых кодов и передачи результатов декодирования в терминал сбора данных или базу данных предприятия.

4.2 Описание структуры приложения Open SourceHH.Mobile

4.2.1 Элементы управления приложения Open SourceHH.Mobile

Описание элементов и областей приложения Open SourceHH.Mobile

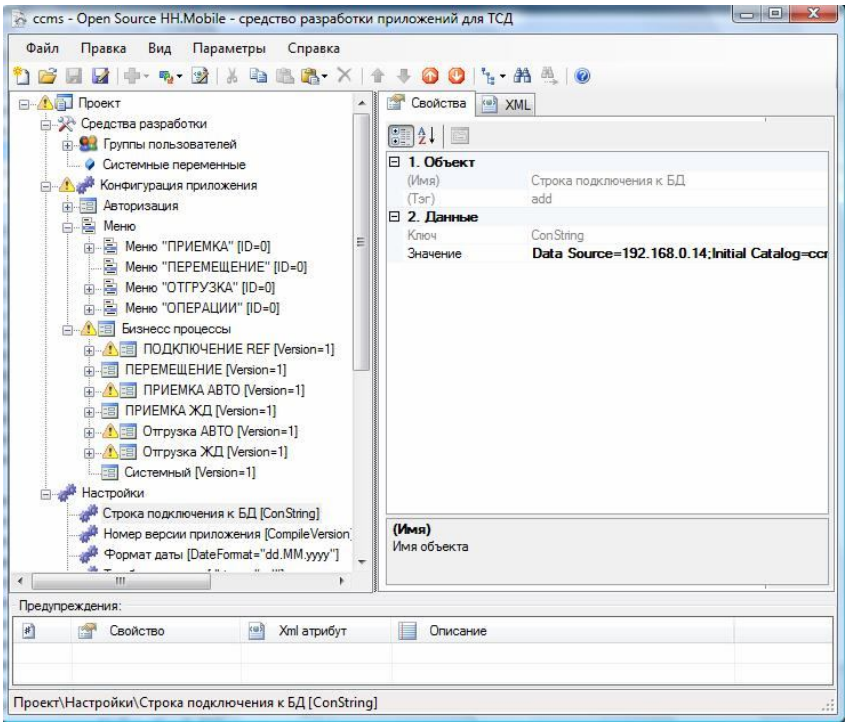
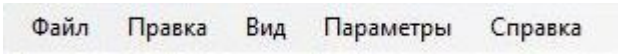


Рис. 4.1. Элементы и области приложения Open SourceHH.Mobile

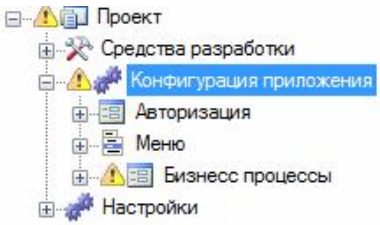
Приложение Open SourceHH.Mobile состоит из следующих областей:

- Функциональное меню. Выбор функциональных действий с объектами проекта и бизнес-процессов.

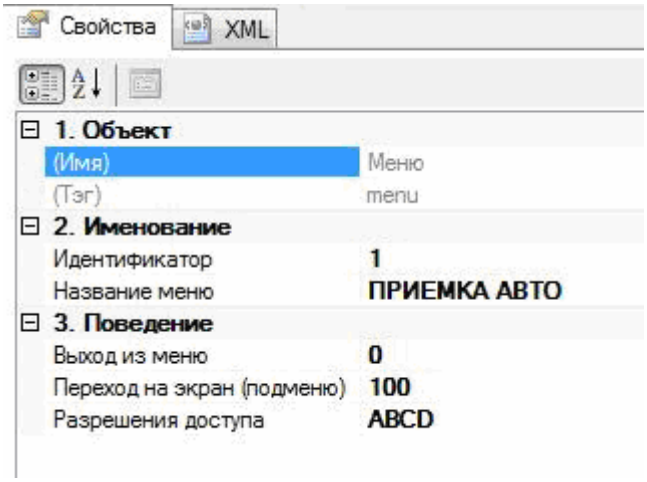


- Панель элементов. Выбор функциональных действий с объектами проекта и бизнес-процессов.

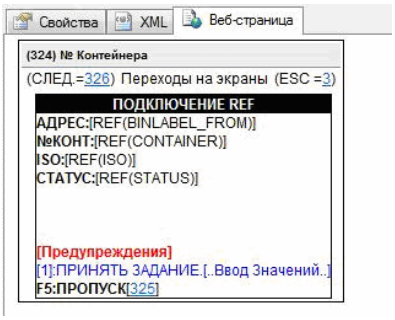
- Дерево проектов. Выбор объектов проекта или бизнес-процессов для просмотра и изменения свойств объектов.



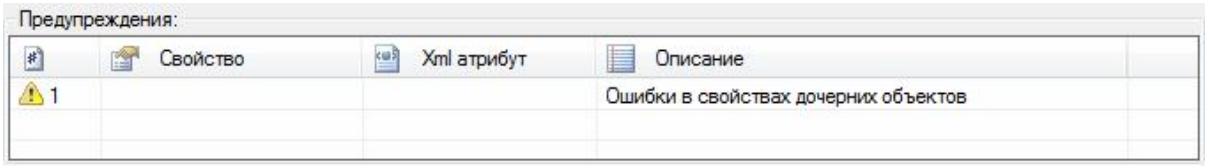
- Окно свойств объектов. Изменение свойств объектов проекта или бизнес-процессов



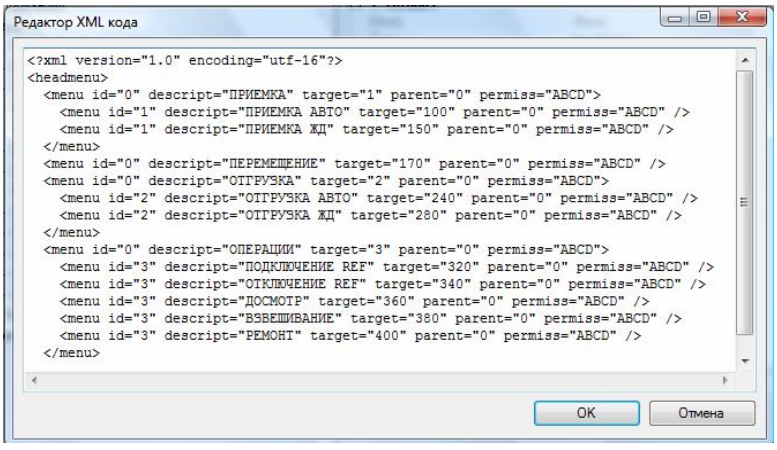
- Окно XML-кода. Просмотр XML-кода объектов проекта или бизнес-процессов
- Окно Веб-страница. Просмотр диаграммы экрана или целиком бизнес-процесса



- Окно предупреждений и ошибок конфигурации



- Окно редактирования XML-кода. Редактирование XML-кода объектов проекта или бизнес-процессов



4.2.2 Настройки приложения Open SourceHH.Mobile

- **Описание групп пользователей системы по категориям**

В конфигурации приложении Open Source HH.Mobile описываются группы пользователей по категориям: администраторы, опраторы и тд.

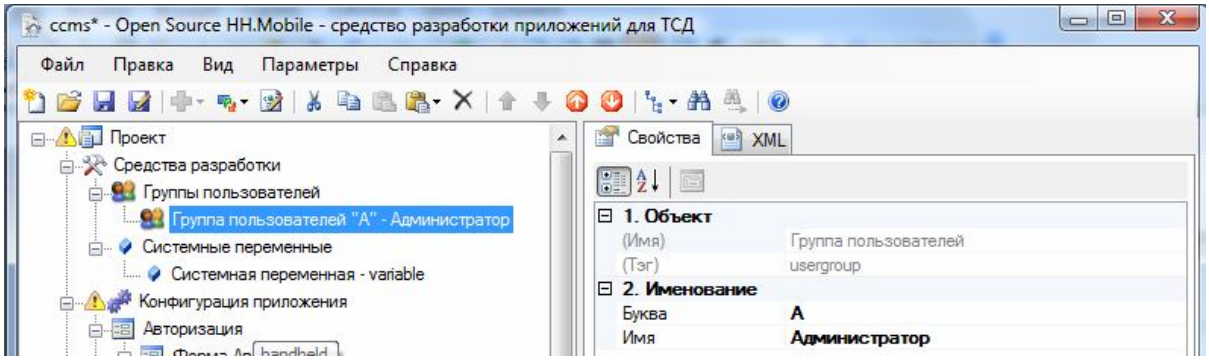


Рис. 4.4. Группы пользователей.

- **Настройка системного файла system.xml**

В конфигурации приложении Open Source HH.Mobile описываются системные настройки сервера приложений, такие как:

- Строка подключения к БД
- Номер версии приложения
- Формат даты
- Тип базы данных
- Номер порта
- Номер удаленного порта
- Максимальное количество строк в меню
- Ширина экрана терминала в символах
- Длина экрана терминала
- Сессия пользователя (сек)
- Значения клавиши подтверждения
- Значение клавиши отмены
- Значение клавиши подсказки
- Значение клавиши пролистывания меню вниз
- Значение клавиши пролистывания меню вверх
- Положение курсора по X
- Положение курсора по Y
- Кодировка терминалов клиента
- Кодировка сервера терминалов

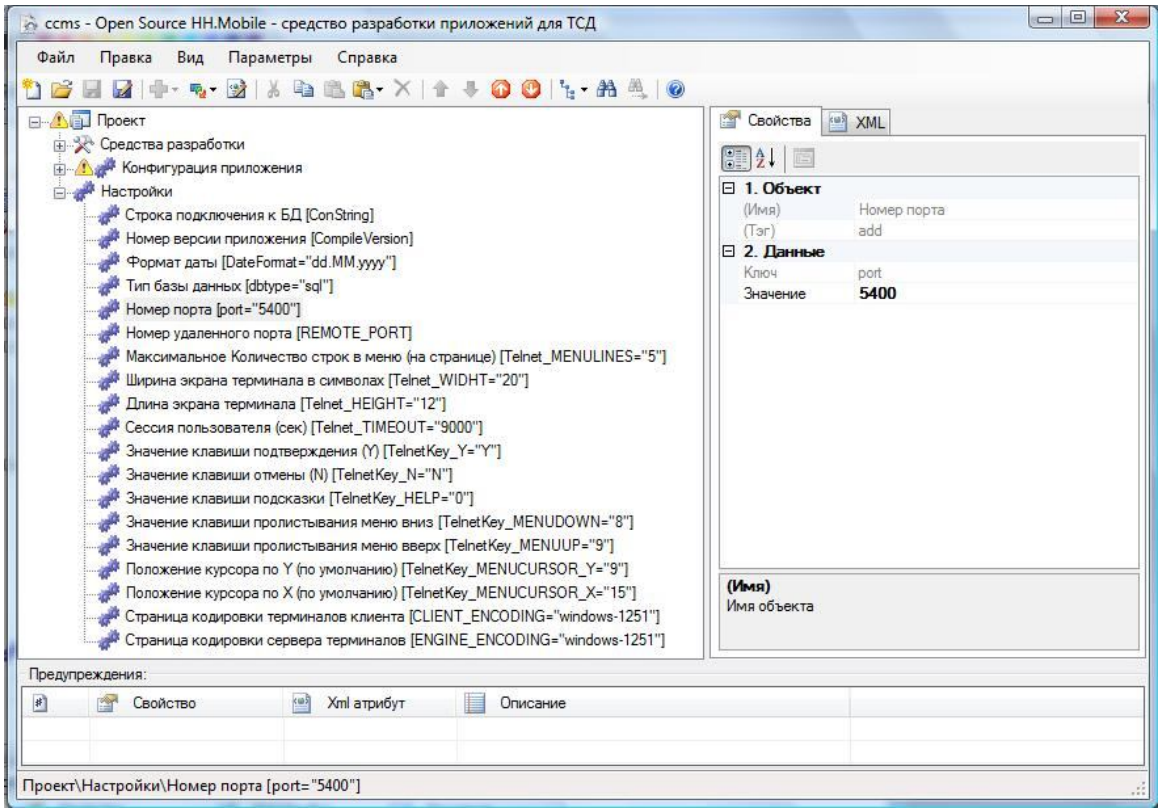


Рис. 4.5. Настройка файла system.xml.

4.2.3 Описание бизнес-процессов в конфигурации приложения

- **Описание окна авторизации пользователей при входе в приложение**

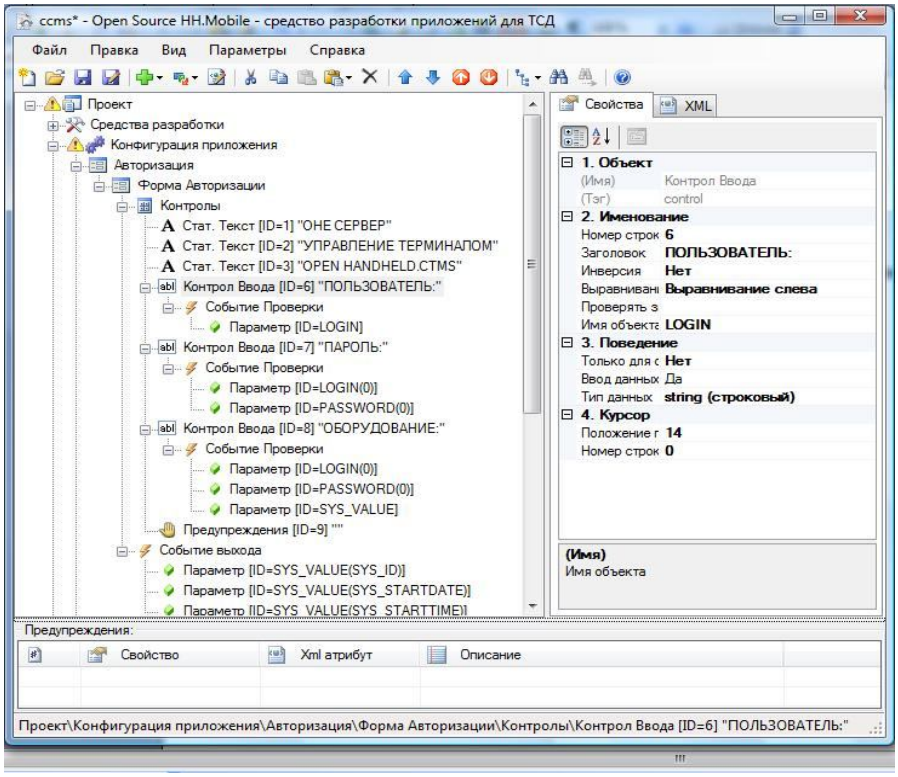


Рис. 4.6. Настройка файла system.xml.

- **Описание пунктов меню приложения**

Конфигурация всех пунктов меню находится в ветке дерева проектов - “Меню”.

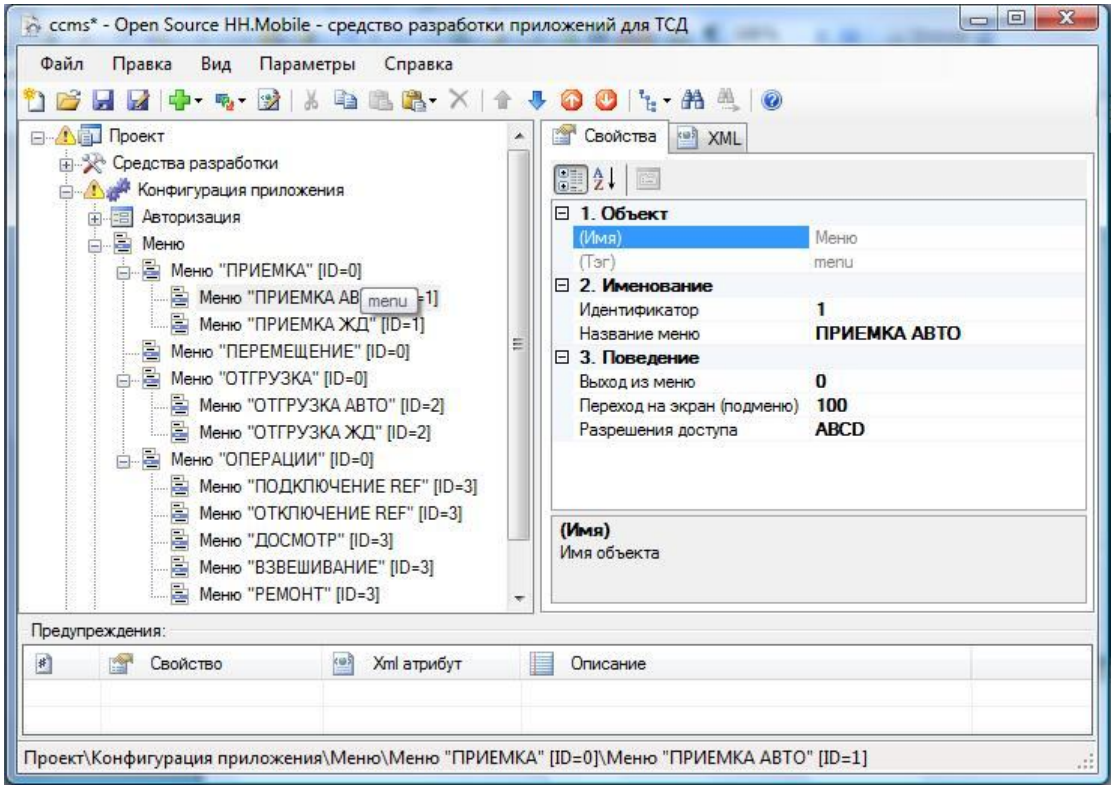


Рис. 4.7. Редактирование пунктов меню из окна свойств.

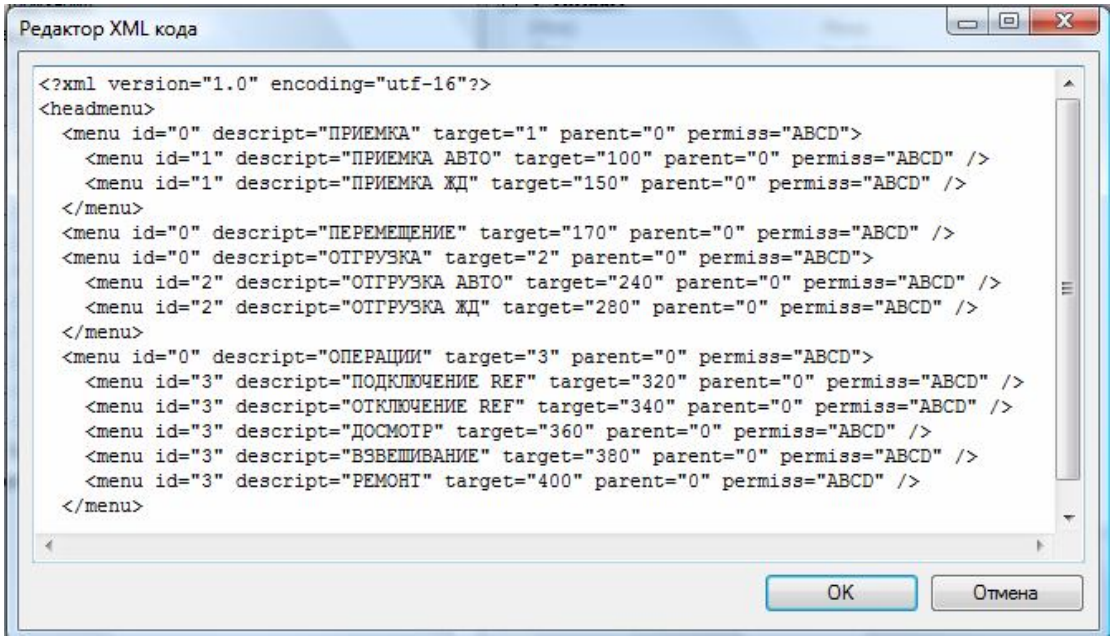


Рис. 4.8. Редактирование пунктов меню из окна редактирования XML кода.

- **Описание бизнес-процессов приложения**

Конфигурация всех разрабатываемых процессов находитсяв ветке дерева проектов - “Бизнес-процессы”.

Каждый процесс состоит из совокупности экранных форм. Каждая экранная форма описывается набором свойств и событий. В приложении Open Source HH.Mobile доступно ограниченное число стандартных и ajax-объектов. По согласованию с компанией “Консид Решения” в Open Source HH.Mobile могут быть добавлены новые объекты.

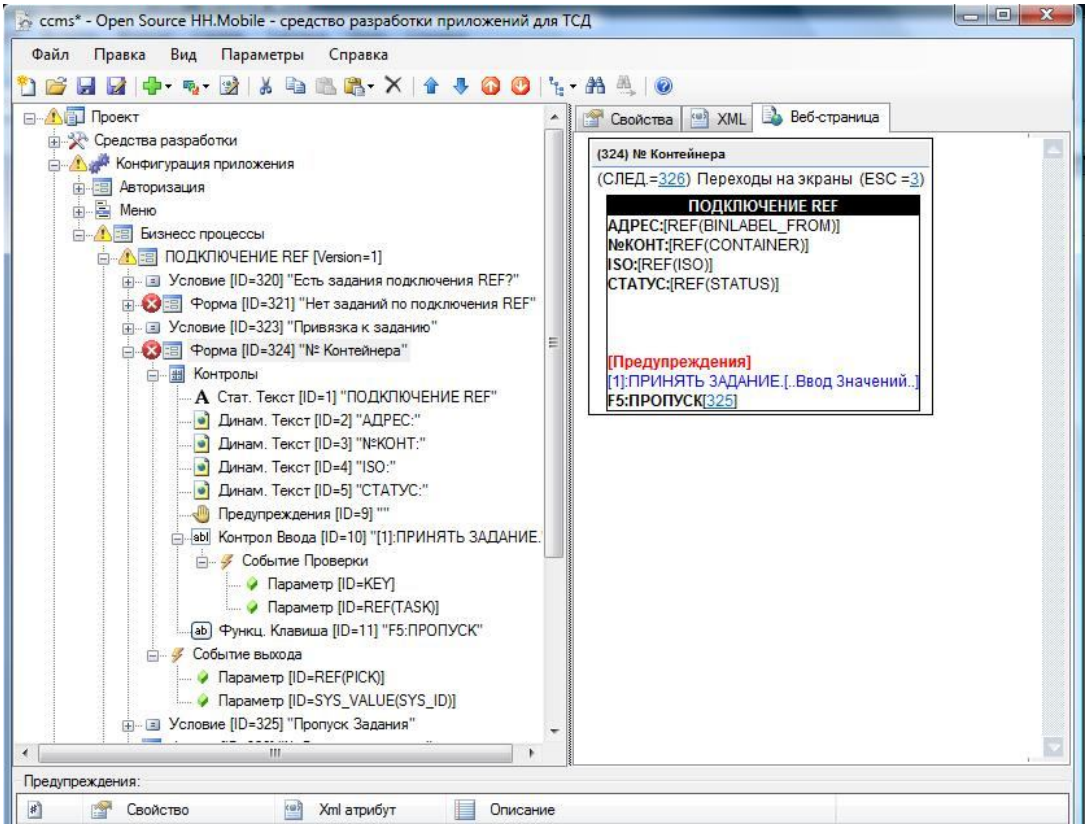


Рис. 4.9. Редактирование бизнес-процесса из окна свойства.

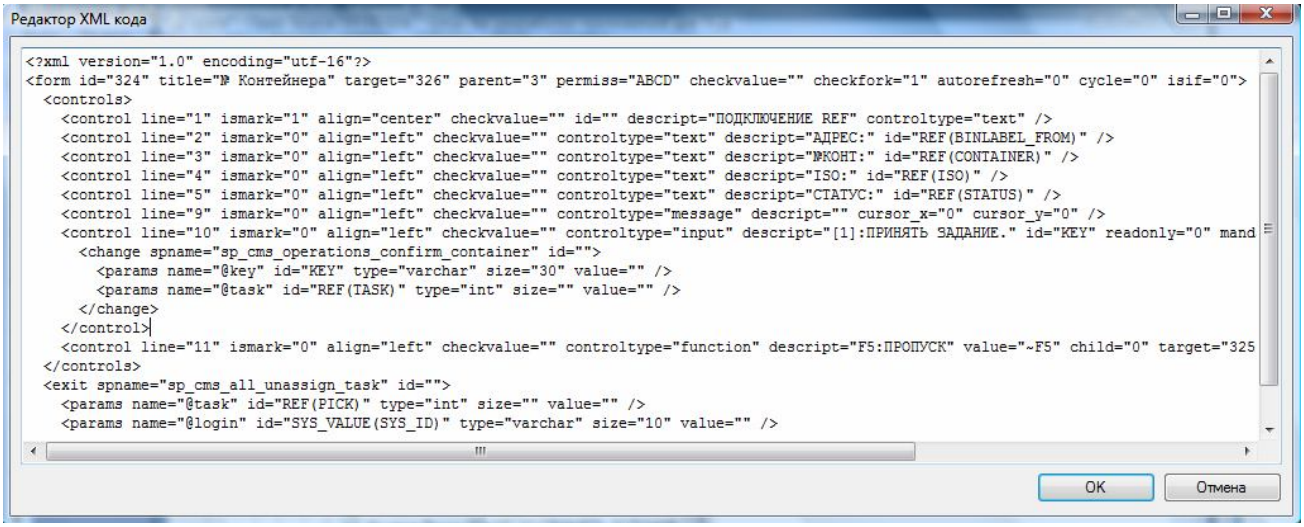


Рис. 4.10. Редактирование бизнес-процесса из окна редактирования XML кода.

4.3 Управляющие элементы и свойства

4.3.1 Описание конфигурации меню

Конфигурация меню описана в файле config.xml, узел headmenu.

Каждый пункт меню в конфигурации описывается узлом – **menu,** который имеет следующие атрибуты, представленные в таблице 4.1. Порядок появления пунктов меню на терминале соответствует порядку появления его в файле конфигурации.

Таблица 4.1. Описание меню Open HandHeld Engine

Название атрибута	Возможные значения	Обязательный / необязательный	Описание
id	Содержит только цифры	Да	Определяет идентификатор меню. Если это меню является главным, то значение равно 0.
descript	Может содержать все символы латинского/русского алфавита	Да	Определяет название меню, как отображается на экране Терминала
permiss	Содержит один или несколько символов латинского алфавита	Да	Определяет привилегию меню. На основе ее определяются все меню и процессы, доступные для пользователей. Каждый символ соответствует привилегии пользователя.
target	Содержит только цифры	Да	Определяет идентификатор экрана терминала или подменю, куда надо перейти при выборе этого меню.
parent	Содержит только цифры	Нет	Определяет куда надо перейти в случае выхода из текущего меню. Не имеет смысла для главного меню.

4.3.2 Описание конфигурации экранных форм

Все экранные формы находятся в узле конфигурации **configuration**. Каждый бизнес-процесс описан в комплексном типе **process**, который содержит список вех форм принадлежащих процессу.

Каждый бизнес-процесс имеет два параметра:

- наименование бизнес-процесса

- версия

При компиляции сервера используется версия бизнес-процесса, описанная в конфигурации приложения.

Формы бывают двух типов:

- Формы экранов терминалов, используются пользователем при выполнении операций.
- Формы условий. Невидимые формы, которые проверяют какие-либо условия, и в зависимости от результатов возвращают пользователя на следующую форму экранов терминалов.

Отличие первого и второго типа форм – атрибут **isif.** Для форм условий должен принимать значение true.

Формат описания двух типов форм одинаков. Единственное отличие, что формы типа 2 должны обязательно иметь событие init формы и не иметь контролов формы.

Если условие выполнилось удачно (@result = 1) пользователь переходит на форму терминалов, или форму условий, определенной в атрибуте **target**. В противном случае пользователь посылается на идентификатор, указанный в атрибуте **parent.**

Каждая экранная форма в конфигурации описывается комплексным типом – **form,** которая имеет следующие атрибуты, представленные в таблице 4.2.

Таблица 4.2. Описание атрибутов экранных форм

Название атрибута	Возможные значения	Обязат-й / необязат-й	Описание
id	Содержит только цифры	Да	Определяет идентификатор формы. Он должен быть **уникальный**.
cycle **Устаревший параметр в дальнейшем не используется**	Может содержать только 0, 1, true, false.	Да	Определяет является ли форма частью бизнес-процесса, выполняющегося в цикле (например приемка по товару). Если значение равно 1, то переход от одной экранной форме к другой, осуществляется без отображения результатов обработки (событие submit). Если 0, то форма должна иметь событие инициализации.
permiss	Содержит один или несколько символов латинского алфавита	Да	Определяет привилегию пользователя на использование экранной формы. На основе ее определяются все процессы, доступные для пользователей. Один символ в атрибуте соответствует привилегии пользователей
target	Содержит только цифры	Да	Определяет идентификатор экрана терминала, куда надо перейти при завершении работы с этой экранной формой. Для форм условий смотри описание выше.
parent	Содержит только цифры	Да	Определяет куда надо перейти (идентификатор формы) в случае выхода из текущей экранной формы. Для форм условий смотри описание выше.
title	Содержит один или несколько символов латинского	Нет	Описание формы, которое видно в конфигураторе

	алфавита		
autorefresh	Может содержать только 0, 1, true, false.	Нет	Признак очистки всех переменных бизнес процесса. По умолчанию переменные не очищаются.
checkvalue	Латинские буквы	Нет	Ссылка на переменную глобального кеша. Если значение этой переменной равно «Y», то экран выводиться, если значение «N» форма не показывается.
isif	Может содержать только 0, 1, true, false.		Переменная определяет, является ли форма формой условий или нет
checkfork	Может содержать только 0, 1, true, false.		Переменная определяет необходимость проверки погрузчика пользователя, если он нажал кнопку выхода с формы. По умолчанию проверки не делаются.
init	Комплексный тип	Нет	Описывает События инициализации экранной формы.
submit	Комплексный тип	Нет	Описывает Событие перехода экранной формы.
exit	Комплексный тип	Нет	Описывает Событие выхода с экранной формы.
controls	Комплексный тип	Да	Описывает все управляющие элементы экранной формы

Любая экранная форма может поддерживать События инициализации, События перехода и выхода из формы.

Пример описания события:

```
<submit spname="sp_updateqty" id=”PONUM”>
    <params name="@id" type="varchar" size="20" id="PROD(0)" />
    <params name="@order_id" type="varchar" size="20" id="PO_NUM(1)" />
    <params name="@qty" type="int" size="20" id="QTY" />
</submit>
```

Описание событий форм и управляющих элементов аналогичны, поэтому будут приведены в одной таблице.

Таблица 4. 3. Описание событий для экранных форм и управляющих элементов

Название атрибута	Возможные значения	Обязат-й / необязт-й	Описание
init **change**	**change** **init**	Нет	Определяет название события, используются три вида событий

submit **exit**	**submit** **exit**		init – событие инициализации change – событие обновления submit – событие перехода exit – событие выхода с формы
spname	Содержит символы латинского алфавита	Да	Содержит название хранимой процедуры, которая вызывается при возникновении события
id	Латинские буквы	нет	Определяет идентификатор таблицы, в которую будут перемещаться данные возвращенные из хранимой процедуры. **Имеет значение только для событий формы.** Не используется для событий контролов.
params	Комплексный тип	Нет	Определяет список параметров, используемых для вызова хранимой процедуры
Описание параметров вызова хранимой процедуры – узел params			
name	Содержит символы латинского алфавита, обязательно начинается со знака «@»	Да	Определяет название параметра.
type	Принимает одно из ниже приведенных значений: **string**–строка **datetime**–дата **int**-целое число **int64**-длинное целое число **decimal**-дробное число **bool**-логическое да нет **byte** - байт **uniqueidentifier** – уникальный идентификатор	Да	Задает тип значения параметра.
size	Число	Нет	Описывает длину параметра, обязателен для текстовых параметров
value	Символы	Нет	Определяет значения параметров для хранимой процедуры
id	Символы	Нет	Определяет идентификатор записи и порядковый номер или название колонки, из которой необходимо брать значение параметра.

4.3.3 Описание управляющих элементов экранных форм

Все управляющие элементы экранных форм должны содержаться в узле controls файла конфигурации. Каждый управляющий элемент должен начинаться с узла “control”.

Все экранные формы обязательно содержат список управляющих элементов, с которыми работает пользователь в процессе выполнения бизнес-процессов.

Все управляющие элементы разделяются на шесть типов:

text - Представляет собой текст, который будет виден на экране терминала. Это статический или динамический текст, который может содержать подсказки или какие-то сообщения которые отображаются на терминале.

input - Контрол ввода пользовательских данных.

Все контролы типа **input** могут поддерживать два события

init - инициализация контрола. Автоматически вызывается при попадании пользователя на форму (если событие определено для контрола).

change - событие обновления контрола. Атоматически вызывается при введении пользователем данных в приложении (если событие определено для контрола).

Кроме этого, ввод данных от пользователя проверяется на соответсвие типу значения контрола.

list - описывает список значений, которые выбираются из этого списка.

function – Управлющий элемент, который определяет наличие функциональных клавиш на экранной форме (например F1…F15, Y, N и т.п.)

help – содержит статический текст, используется для подсказок на экранных формах, при нажатии на клавишу помощи.

message – содержит данные, возвращенные хранимой процедурой, обычно описание ошибок, или исключений.

Контрол типа text

Элемент управления типа text подразделяется на два типа – статический и динамический – определен аттрибут id. Динамический текст отображает значение в одной из колонок таблицы, которую возвращает хранимая процедура.

Статический - нет определяния атрибута id. Это просто текстовая информация, отображаемая на экране терминала.

Контролы данного типа не поддерживают никаких событий.

Описание возможных атрибутов описана ниже в таблице.

Таблиц 4.4. Описание атрибутов контрола типа text

Название атрибута	Возможные значения	Обязательный / необязальный	Описание
line	Содержит только цифры	Да	Определяет номер строки, на которой отображается контрол на экранной форме.
controltype	text	Да	Определяет тип управляющего элемента
descript	Любой текст	Нет	Задает статический текст который отображается на терминале
align	Возможные значения: left – слева right - справа center – по центру justify – равномерно по строке	Нет	Определяет выравнивание управляющего элемента по горизонтали. Если не задан используется значение left.
value	Любой текст	Нет	Задает статический текст который отображается на терминале
id	Любой текст	Нет	Используется для динамических текстовых контролов.
ismark	Может содержать только 0, 1 или true, false.	Нет	По умолчанию, 0 – контрол не выделен. Если значение установлено в true, тогда у контрола меняется фон.
checkvalue	Латинские буквы	Нет	Ссылка на переменную глобального кеша. Если значение этой переменной равно «Y», то экран выводиться, если значение «N» форма не показывается.

Примеры определения конфигурации контрола.

Определение статического текста:

```
<control
        controltype="text"
        line ="3"
        descript="Состав волны"
        ismark="1">
</control>
```

Описание – это текстовая строка с выравниванием по левому краю, находящаяся на третьей строке терминала и отображает текст “Состав волны”. Фон контрола – инверсия на терминале.

Определение динамического текста.

```
<control
        controltype="text"
        line ="4"
        descript="Заказы:"
        align ="right"
        id="PO_NUM(ORDERS)">
</control>
```

Описание – это текстовая строка с выравниванием по правому краю, находящаяся на четвертой строке терминала и отображает текст “Заказы:” + значение из таблицы PO_NUM, колонка ORDERS.

Контрол типа input

Контрол типа **input** представляет собой контрол для ввода данных пользователем.

Управляющий элемент данного типа может поддерживать два события:

- **init** – событие, которое возникает при первом отображении экранной формы на терминале.

- **change** – событие, возникающее при вводе данных пользователем.

Описание конфигурации событий см. пункт “Описание событий управляющих элементов”

Описание возможных атрибутов описана ниже в таблице.

Таблиц 4.5. Описание атрибутов контрола типа input

Название атрибута	Возможные значения	Обязат-й / необязат-й	Описание
id	Латинские буквы	Да	Идентификатор контрола, определяет название таблицы при наличии событий, или идентификатор значения контрола при отсутствии событий.
controltype	input	Да	Определяет тип управляющего элемента
line	Содержит только цифры	Да	Определяет номер строки, на которой отображается контрол на экранной форме.
type	Принимает одно	Да	Определяет тип значений контрола

	из значений: **string**–строка **datetime**–дата **int**-целое число **int64**-длинное целое число **decimal**-дробное число **bool**-логическое да нет **byte** - байт **uniqueidentifier** – 32 разрядный уникальный идентификатор MSSQL сервер		
mandatory	Может содержать только 0, 1 или true, false.	Нет	Определяет, является обязательным для ввода или нет. Если значение не задано, то возможно пустое значение.
readonly	Может содержать только 0, 1 или true, false.	Нет	Определяет, является ли контрол для чтения или ввода данных. Если значение не задано, то контрол для ввода данных
descript	Любой текст	Нет	Задает статический текст, который отображается на терминале
value	Любой текст	Нет	Задает статический текст, который отображается на терминале
cursor_y	Цифра	Нет	Определяет положение курсора по вертикали для ввода значений. Значение курсора рассчитывается относительно значения атрибута line. Т.е. номер строки, на которой будет находится курсор равен line + cursor_y. Если значение не определено, то считается что курсор находится на той же строке что и контрол.
cursor_x	Цифра	Нет	Определяет положение курсора ввода по горизонтали. Если значение не задано, то определяется исходя из длины атрибута descript + 1

Примеры определения конфигурации контрола.

Определение контрола ввода текста:

```
<control id="PROD"
        controltype="input"
        descript="TOBAP:"
        type="string"
        readonly="0"
        mandatory="1"
        line="4"
        cursor_y="4"
        cursor_x="2">
                <init spname="sp_getproduct">
                        <params
                                name="@order_id"
                                type="varchar"
                                size="20"
                                id="PO_NUM(1)">
                        </params>
                </init>
                <change spname="sp_product_exists">
                        <params
                                name="@product_id"
                                type="varchar"
                                size="20"
                                id="PROD(0)">
                        </params>
                        <params
                                name="@order_id"
                                type="varchar"
                                size="20"
                                id="PO_NUM(1)">
                        </params>
                </change>
</control>
```

Управляющий элемент ввода данных пользователем расположен на 4 строке терминала, причем курсор для ввода данных находится на 8 строке и второй позиции в строке. Контрол имеет два события – инициализации и изменения. При инициализации контроляа вызывается процедура с одним **sp_getproduct** параметром **PO_NUM(1)** , где 1- порядковый номер колонки, начиная с 0. Ввод данных пользователем обязателен. После

ввода данных пользователем, происходит проверка типа, и вызывается хранимая процедура **sp_product_exists** которая обрабатывает введенные данные и возвращает результат, в зависимости от которого пользователь переходит (или нет) на следующий экран или к другому контролу.

Определение контрола ввода количества:

```
<control
        id="QTY"
        line="5"
        controltype="input"
        descript="КОЛ-ВО:"
        type="int"
        readonly="0"
        mandatory="1"
        cursor_y="3"
        cursor_x="2">
</control>
```

Управляющий контрол ввода количества. Расположен на 5 строке экрана терминала. Описание контрола – «КОЛ-ВО», обязательный для ввода данных пользователем, не имеет событий, но проверяется на ввод данных типа int. Курсор для ввода расположен на 8 строке со второй позиции.

Контрол типа list

Контрол данного типа представляеяет собой пользовательский список данных с возможностью выбора одного из многих значений списка. Это подобие выпадающего меню. При появлении этого управляющего элемента, пользователь обязательно должен выбрать одно из значений.

Данный управляющий контрол может содержать только одно событие – инициализации.

Таблиц 4.6. Описание атрибутов контрола типа list

Название атрибута	Возможные значения	Обязат-й / необязат-й	Описание
id	Латинские буквы	Да	Идентификатор контрола, определяет название таблицы при наличии событий, или идентификатор значения контрола при отсутствии событий.
controltype	list	Да	Определяет тип управляющего элемента
line	Содержит только цифры	Да	Определяет номер строки, на которой отображается контрол на экранной форме.
cursor_y	Цифра	Нет	Определяет положение курсора по вертикали для ввода значений. Значение курсора рассчитывается относительно

			значения атрибута line. Т.е. номер строки, на которой будет находится курсор равен line + cursor_y. Если значение не определено, то считается что курсор находится на той же строке что и контрол.
cursor_x	Цифра	Нет	Определяет положение курсора ввода по горизонтали. Если значение не задано, то определяется исходя из длины атрибута descript + 1
checkvalue	Латинские буквы	Нет	Ссылка на переменную глобального кеша. Если значение этой переменной равно «Y», то экран выводиться, если значение «N» форма не показывается.

Примеры определения конфигурации контрола:

```
<control
    id="PO_NUM"
    controltype="list"
    readonly="0"
    mandatory="1"
    line="3"
    cursor_y="-1"
    cursor_x="0">
        <init
            spname="select_avail_orders">
        </init>
    </control>
```

или

```
<control
    id="PO_NUM"
    controltype="list"
    readonly="0"
    mandatory="1"
    line="3"
    cursor_y="-1"
    cursor_x="0">
      <option id ="1" value ="PO1"/>
      <option id ="2" value ="PO2"/>
      <option id ="3" value ="PO3"/>
</control>
```

Описание атрибутов подобно остальным контролам. Единственное отличие в том, что список не имеет событий обновления, но должен иметь или событие инициализации, или список предопределенных параметров (см. второй пример). Еще одной из особенностей является то, что контрол не предусматривает ввод пустого значения. Т.е. пользователь обязательно должен выбрать одно из значений в списке.

Контрол типа function

Контрол данного типа – это аналог использования горячих клавиш windows. Используется в соновном для вывода списков даннных и перехода из одного бизнесс-процесса в другой. Контрол реагирует на нажатие клавиши пользователем и пересылает его на форму, определенную в конфигурации. Может поддерживать любые клавиши, включая функциональные.

Атрибуты управляющего контрола function представлены в таблице.

Таблиц 4.7. Описание атрибутов контрола типа function

Название атрибута	Возможные значения	Обязательный / необязальный	Описание
controltype	function	Да	Определяет тип управляющего элемента
line	Содержит только цифры	Да	Определяет номер строки, на которой отображается контрол на экранной форме.
value	Любой текст	Да	Определяет клавишу, при нажатии на которую активируется контрол.
descript	Любой текст	Нет	Задает статический текст, который отображается на терминале. Задает пользовательское описание контрола.
target	Число	Да	Определяет идентификатор формы, куда надо перейти при нажатии клавиши, указанной в атрибуте **value.**
child	true,false	Да	Определяет, является ли вызываемая форма подчиненной или нет.
checkvalue	Латинские буквы	Нет	Ссылка на переменную глобального кеша. Если значение этой переменной равно «Y», то экран выводиться, если значение «N» форма не показывается.

Пример определения конфигурации контрола:

```
<control
        controltype="function"
        line ="9"
        value="~F5"
        target ="15"
        descript="СПИСОК: F5"
        child=”true”>
</control>
```

Управляющий элемент находиться на 9 строке терминала и отображаемая строка – «СПИСОК: F5» и при нажатии клавиши F5 сработает вызов фомы с идентификатором 15 и управление передасться ей. Атрибут **child = true** показывает что вызываемая форма является подчиненной и результат ее выполнения должен передаться в текущий активный контрол.

Контрол типа help

Данный контрол представляет из себя пользовательскую подсказку, которая описывает работу с текущим экраном. При нажатии функциональной клавиши, появляется меню, в котором описан списк возможный пользовательских действий для работы с текущей формой. На каждую форму можно пдоключить один контрол данного типа.

Пример описания контрола.

Контрол типа message

Контрол данного типа возращает результат обработки данных хранимой процедурой, в том случае если процедура вернула отрицательный результат, или пользователь ввел некорректные данные. Используется для интерактивной работы с пользователем при вводе даннных. Рекомендуется использовать на каждом экране.

Пример описания контрола.

```
<control controltype="message" line ="5" cursor_y ="0" cursor_x ="2" />
```

Данные контрол будет отображаться на 5 строке терминала и результат будет выводиться на той-же строке сос второй позиции.

4.3.4 Описание событий управляющих элементов

Управляющие события, могут иметь только контролы, которые связаны с обработкой данных введенных пользователем. Т.е. это контролы типа **input** и **list**.

При чем, контрол типа **input** может иметь необязательные события инициализации и обновления.

А контрол типа **list** может иметь только необязательное событие инициализации.

Событие инициализации возникает при загрузке формы на экран в первый раз. А событие обновления активируется после введения пользователем данных.

Оба события должны возвращать код, который соответствует успешной или неудачной обработке события.

Событие инициализации описывается комплексным типом **init**, который имеет один атрибут – **spname** (название хранимой процедуры), и комплексным типом **params** – который описывает имя, тип, значение параметров вызова процедуры.

Аналогично описывается событие обновления, только вместо комплексного типа init используется тип **change**.

Ниже представлены атрибуты параметров событий.

Таблиц 4.8. Описание атрибутов контрола типа text

Название атрибута	**Возможные значения**	**Обязат-й / необязат-й**	**Описание**
name	Тескт, который начинается со знака «@». Действуют правила описания параметров хранимых процедур сервера БД	Да	Определяет название параметра.
type	Описывает тип параметра. Используются типы, которые поддерживаются базой данных	Да	Определяет тип параметра.
size	Число	нет	Определяет длину значения параметра, имеет смысл только для текстовых параметров.
id	Любой текст	Нет	Определяет идентификатор таблицы и имя колонки, где надо взять значение параметра.
value	Число	Да	Определяет статическое значение параметра, если это константа.

Пример описания события инициализации.

```
<init
        spname="sp_getproduct">
         <params
                name="@order_id"
                type="varchar"
                size="20"
                id="PO_NUM(1)">
         </params>
</init>
```

При возникновении события вызывается процедура **sp_getproduct,** в которую передается один параметр **@order_id** типа varchar(20). Значение для этого параметра берется из таблицы **PO_NUM,** из второй колонки (порядковый номер колонок в таблице начинается с 0).

Пример описания события обновления.

```
<change
        spname="sp_product_exists">
         <params
                name="@product_id"
                type="varchar"
                size="20"
                id="PROD(0)">
         </params>
         <params
                name="@type_id"
                type="int"
                value="1" >
         </params>
</change>
```

При вводе данных пользователем вызывается хранимая процедура **sp_product_exists,** которая имеет два параметра.

Параметр **@product_id** – тип varchar(20) и значение берется из таблицы PROD первой колонки. Второй параметр **type_id** – это число со значением 1.

Использование параметров в событиях необязательно. Пример вызова процедуры без параметров, например для события инициализации ниже.

```
<init  spname="sp_getallproduct"> </init>
```

5 Требование к разработке хранимых процедур

5.1 Используемые термины и сокращения

Телнет Сервер или Сервер - Программный Комплекс, обеспечивающий взаимодействие между пользователями системы, работающими с Терминалами и Сервером, содержащим ядро для разработки бизнес-процессов клиента.

Сервер БД – Компьютер, предназначенный для хранения и обработки данных, поступающих от Модулей и АРМ.

Сервер – часть Программы, находящаяся на персональном компьютере и необходимая для обмена данными между компьютером и Терминалом.

Хранимая процедура – кусок кода на языке SQL, содержащий логику работы с базой данных, вызывается из приложения Телнет Сервер, храниться на Сервере БД.

Переменные сессии пользователя – набор данных (ключ, значение), которые определяются при авторизации пользователя в системе и хранятся в течение всего сеанса работы.

Предопределенные глобальные переменные – набор данных (ключ, значение), которые определяются в момент подключения пользователя к Телнет Серверу. Пользователь не может добавлять этот тип переменных, но может использовать.

Переменные процесса – набор данных, который определяется при выполнении каких-либо бизнес операций, например при приемке – определяется номер заказа прихода. Эти данные существуют только когда пользователь выполняет какую-либо операцию.

Ключ – уникальный идентификатор переменной, который используется для определения и задания параметров данных, каждый ключ может иметь значение.

Значение – набор данных, которые хранятся в переменной с уникальным идентификатором – ключом. Значения может быть как скалярным, так и таблицей.

5.2 Требование к разработке хранимых процедур

Все хранимые процедура используемые при работе Телнет Сервера должна удовлетворять следующим требованиям:

1. Всегда должен быть определен выходной параметр (@result(bit)), который принимает значения 1 – если хранимая процедура выполнилась успешно, 0 – в противном случае. Данные параметр следует добавлять только в хранимую процедуру. *Добавление этого параметра в конфигурацию приведет к ошибке запуска хранимой процедуры*.

2. Всегда должна возвращать значения в виде таблицы (использовать select), а том случае если хранимая процедура выполнилась неуспешно, то возвращать описание ошибки. Таблица должна содержать как минимум одну строку.

3. Возможно генерирование ошибок SQL сервера. Эти ошибки будут перехвачены приложением, а описание ошибки запишется в лог.

4. Если необходимо в значениях использовать перевод строки, то надо добавить выражение «{0}» в результат вывода. Данное выражение автоматически замениться на символ возврата каретки.

При разработке хранимых процедур следует стремиться, чтобы процедура возвращала одну строку (за исключением процедур, заполняющих элемент – «список»). Это исключает необходимость дополнительных действий пользователя по вводу данных.

Если не получается возвратить из хранимой процедуры одну строку (н-р, несколько кодов товара имеют один штрих код), система отловит эту ситуацию и динамически выведет форму со списком, из которого необходимо выбрать одно значение.

Для этих хранимых процедур следует руководствоваться следующими принципами при возврате данных:

1. Первая колонка таблицы должна содержать уникальный код параметра.
2. Вторая колонка должна содержать описание параметра – именно оно будет динамически выводиться на экран в виде списка.

5.2.1 Предопределенные глобальные переменные

Телнет Сервер использует некоторое количество предопределенных глобальных переменных, которые используются как для системных вещей, так и в качестве параметров хранимых процедур.

Список переменных:

SYS_ERROR – содержит описание текущей ошибки, которая возращается из хранимой процедуры. Используется управляющим элементом типа **message**.

SYS_DATE – содержит текущую дату, описанную в конфигурации приложения.

SYS_DATETIME – содержит текущую дату и время в формате, описанном в конфигурации приложения.

SYS_TIME - содержит текущее время в формате HH:mm.

SYS_YEAR – содержит текущий год.

SYS_MONTH – содержит текущий месяц.

SYS_DAY - содержит текущий день.

Примечание: *В целях более логичного использования переменных сессии пользователя рекомендуется использовать префикс "SYS_". Это гарантирует, что переменные этого типа не удаляться до момента окончания сеанса пользователя.*

Все переменные без префикса «SYS_» удаляются из хранилища, если пользователь вышел из процесса в меню.

5.2.2 Рекомендации по названию переменных в конфигурации

Все ключи переменных должны удовлетворять следующим требованиям:

1. Содержать только латинские буквы. Если в ключе переменной используется только одна из колонок для вывода значения, то она должна заключаться в круглые скобки.
2. Ключи переменных должны быть уникальны на уровне процесса (если используются переменные процесса)
3. По возможности использовать смысловые имена переменных.
4. Использовать префикс «SYS_» только для переменных сессии пользователя

Все переменные сессии пользователя и глобальные переменные должны начинаться с префикса «SYS_». Использование этого префикса гарантирует постоянное значение переменной в течении сессии пользователя.

Приложение

Пример файла конфигурации Сервера.

```
<?xml version="1.0" encoding="utf-8" ?>
<handheld>
<users>
  <user id="USER1" password="1" permiss="A" />
  <user id="USER2" password="2" permiss="B" />
  <user id="USER3" password="3" permiss="A" />
  <user id="USER4" password="4" permiss="B" />
  <user id="USER5" password="5" permiss="A" />
  <user id="USER6" password="6" permiss="B" />
</users>
<headmenu>
  <menu id="0" descript="Приемка" permiss="A" target="10" />
  <menu id="0" descript="Отбор" permiss="A" target="3" />
  <menu id="0" descript="Отгрузка" permiss="A" target="4" />
  <menu id="0" descript="Склад" permiss="A" target="5" />
  <menu id="0" descript="Супервизор" permiss="A" target="8" />
<menu id="0" descript="Ячейки" permiss="A" target="20" />
  <menu id="0" descript="Настройки" permiss="A" target="20" />
  <menu id="2" descript="По заказу" permiss="A" target="10" parent="0" />
  <menu id="2" descript="По волне" permiss="A" target="20" parent="0" />
  <menu id="2" descript="От поставщика" permiss="A" target="20" parent="0" />
  <menu id="2" descript="ASN" permiss="A" target="20" parent="0" />
  <menu id="2" descript="КонтейнернаЯ" permiss="A" target="20" parent="0" />
</headmenu>
<forms>
<form id="12" target="13" permiss="AC" parent="10" cycle="0">
<init spname="sp_getproduct">
   <params name="@order_id" type="varchar" size="20" value="" id="PO_NUM(1)" />
</init>
<controls>
<control controltype="text" line="1" descript="ПРИЕМКА ПО ЗАКАЗУ" align="center" />
<control id="PO_NUM(0)" controltype="text" line="2" descript="ПОСТАВКА:" align="center" />
<control id="PROD" controltype="input" descript="ТОВАР:" type="string" readonly="0" mandatory="1"
line="4" cursor_y="4" cursor_x="2">
<init spname="sp_getproduct">
  <params name="@order_id" type="varchar" size="20" value="" id="PO_NUM(1)" />
  </init>
```

```
<change spname="sp_product_exists">
<params name="@product_id" type="varchar" size="20" value="" id="PROD(0)" />
<params name="@order_id" type="varchar" size="20" value="" id="PO_NUM(1)" />
</change>
 </control>
 <control controltype="message" line="6" cursor_y="0" cursor_x="2" />
  <control controltype="text" line="7" descript="СКАНИРУЙ ТОВАР" align="center" />
 <control controltype="text" line="8" descript="[ ]" align="justify" type="string" />
 </controls>
 </form>
 </forms>
</handheld>
```