На правах рукописи

Козак Николай Владимирович

Повышение эффективности процесса разработки систем управления промышленной электроавтоматикой на основе интеграции внешних программных компонентов

Специальность 05.13.06 – Автоматизация и управление технологическими процессами и производствами (технические системы)

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

кандидата технических наук

Москва 2008 г.

Работа выполнена в ГОУ ВПО Московском государственном технологическом университете «СТАНКИН»

Научный руководитель -	доктор технических наук, доцент Мартинов Георги Мартинов
Официальные оппоненты:	доктор технических наук, профессор Аршанский Михаил Маркович кандидат технических наук Акулин Дмитрий Александрович
Ведущая организация:	Научно-исследовательский и экспериментальный институт автомобильной электроники и электрооборудования

Защита диссертации состоится «_____» _____ 2008 г. в _____ часов на заседании диссертационного совета Д 212.142.03 в ГОУ ВПО Московском государственном технологическом университете «СТАНКИН» по адресу: 127994, Москва, Вадковский переулок, д. 3а.

С диссертацией можно ознакомиться в библиотеке ГОУ ВПО Московского государственного технологического университета «СТАНКИН».

Автореферат разослан «_____»_____ 2008 г.

Ученый секретарь
диссертационного Совета,
кандидат технических наук Семячкова Е.Г.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Увеличение сложности процессов автоматизации в промышленности ведут к росту требований, предъявляемых к системам управления электроавтоматикой. Поэтому круг задач, решаемых прикладной составляющей, которые наиболее привлекательны для конечных пользователей, развивается наиболее динамично. С другой стороны существенный прогресс в развитии вычислительной, коммуникационной аппаратуры, в совершенствовании инструментальных средств и технологий разработки программного обеспечения создают предпосылки для решения качественно новых задач в области электроавтоматики.

Анализ архитектуры прикладной составляющей систем электроавтоматики ведущих фирм разработчиков (Siemens, Bosch, Beckhoff), в задачах, их функциональных возможностях и открытости решений выявили следующие проблемы:

- отсутствие единого подхода к систематизации прикладных компонентов приложений систем управления, что усложняет анализ, проектирование и разработку при создании компонентов системы с нуля или расширения её возможностей;
- отсутствие единой глобальной концепции интеграции компонентов от различных производителей в состав приложений разрабатываемой прикладной составляющей системы;
- не специфицированы программные интерфейсы, обеспечивающие более эффективную и глубокую интеграцию функциональности новых компонентов в состав системы;
- не предоставляется возможность гибкого конфигурирования набора прикладных компонентов для создания проблемно-ориентированных решений и настройки под задачи конкретного пользователя.

В результате исследования было определено, что наиболее прогрессивной архитектурой прикладной составляющей системы

управления, обладает открытая система, на основе узкоспециализированных прикладных компонентов, как собственных, так и сторонних производителей.

Исходя из сказанного, тема диссертации направленная на повышение эффективности процесса разработки для прикладной составляющей систем управления промышленной электроавтоматикой на основе интеграции внешних программных компонентов является актуальной.

Цель работы. Повышение эффективности процесса разработки прикладной составляющей промышленной электроавтоматики на основе его формализации, создания единого информационного окружения выполнения и обеспечения интеграции внешних программных компонентов.

Задачи исследования. Для достижения цели в работе были поставлены следующие задачи:

исследовать состав компонентов электроавтоматики и выявить критерии определения минимального набора компонентов на стадии проектирования системы;

на основе принципов классификации разработать методологические аспекты проектирования и разработки систем управления с целью формирования единого исполняемого окружения, для взаимодействия программных компонентов системы;

разработать механизмы интеграции прикладных программных компонентов внешних производителей;

в соответствии с полученными теоретическими и практическими знаниями разработать средство трёхмерной визуализации объекта управления, решить задачи управления доступом и настройки конфигурации для прикладных приложений выбранной системы управления промышленной электроавтоматикой.

Методы исследования. Теоретические исследования в работе базировались на методах системного анализа, объектно-ориентированного проектирования (декомпозиции, абстракции), концепции объектно-

ориентированного программирования. Использовались технологии .Net (для Windows), DCOM (distributed component object model), автоматизация OLE (object linking and embedding), стандарт OPC (OLE for process control).

Научная новизна. Новыми научными результатами, полученными в работе, являются:

- принцип классификации прикладных компонентов электроавтоматики, позволяющий создать матрицу для анализа существующих решений и определения минимальных и оптимальных наборов комплектации компонентов в реализации конкретных пользовательских задач;
- метод проектирования и разработки прикладных приложений электроавтоматики с использованием готовых решений внешних производителей;
- принцип трехуровневого конфигурирования, позволяющий осуществлять настройку прикладных приложений под конкретные производственные потребности и настраивать интерфейс оператора;
- информационно-программное окружение для функционирования прикладных приложений электроавтоматики.

Практическая ценность работы заключается в:

- методике разработки прикладных компонентов и интеграции готовых модулей, позволяющей синтезировать прикладные приложения на основе готовых модулей;
- разработанных программных компонентах для трехмерной визуализации объекта управления и для управления правами пользователей и для конфигурирования прикладных приложений в системе CoDeSys;

позволяющих сократить время и себестоимость процесса разработки и повысить его эффективность.

Апробация работы. Теоретические и практические результаты, полученные автором, докладывались на заседаниях кафедры «Компьютерные

системы управления», а также на международных научно технических конференциях «Информационные средства и технологии», Москва 2004, 2005, 2006 и 2007 г и «Ежегодная XVII Международная Интернет-конференция молодых ученых и студентов по современным проблемам машиноведения» Москва 2005 г.

Публикации. По теме диссертационной работы опубликовано 8 печатных работ.

Структура и объём диссертации. Диссертационная работа состоит из введения, четырёх глав, заключения, выводов по диссертационной работе, списка использованной литературы из 75 наименований. Основная часть работы изложена на 164 страницах машинописного текста, содержит 58 рисунка и 8 таблиц.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обосновывается актуальность работы, отмечена её научная новизна и практическая ценность.

В первой главе, на основе научных трудов Сосонкина В.Л., Мартинова Г.М., Фельдмана Я.А. и других специалистов в области разработки программного обеспечения систем управления и информационных систем в промышленности, производится анализ систем управления электроавтоматикой с позиций требований, предъявляемых к ним. Исследованы архитектуры организации приложений и современный процесс разработки для прикладной составляющей систем управления электроавтоматикой.

Тенденции построения прикладного программного обеспечения систем управления заключаются в: использовании технологических и стоимостных преимуществ персональных компьютеров; предоставления современных средств разработки пользовательского интерфейса прикладных приложений, применения объектно-ориентированного и компонентного подходов в программной реализации; предоставления возможности интеграции со

SCADA, MES и ERP системами. Программное обеспечение систем управления строится по принципам открытой модульной архитектуры.

В результате проведённого анализа выделены три типа требований предъявляемых к прикладным компонентам электроавтоматики: со стороны ядра системы управления, со стороны конечных пользователей и со стороны технологии разработки. Первые два типа требований формируют функциональные возможности компонентов. Последний тип характеризует процесс разработки. При разработке должна учитываться как специфика области реализации, так и возможность применения новых подходов в создании программного обеспечения.

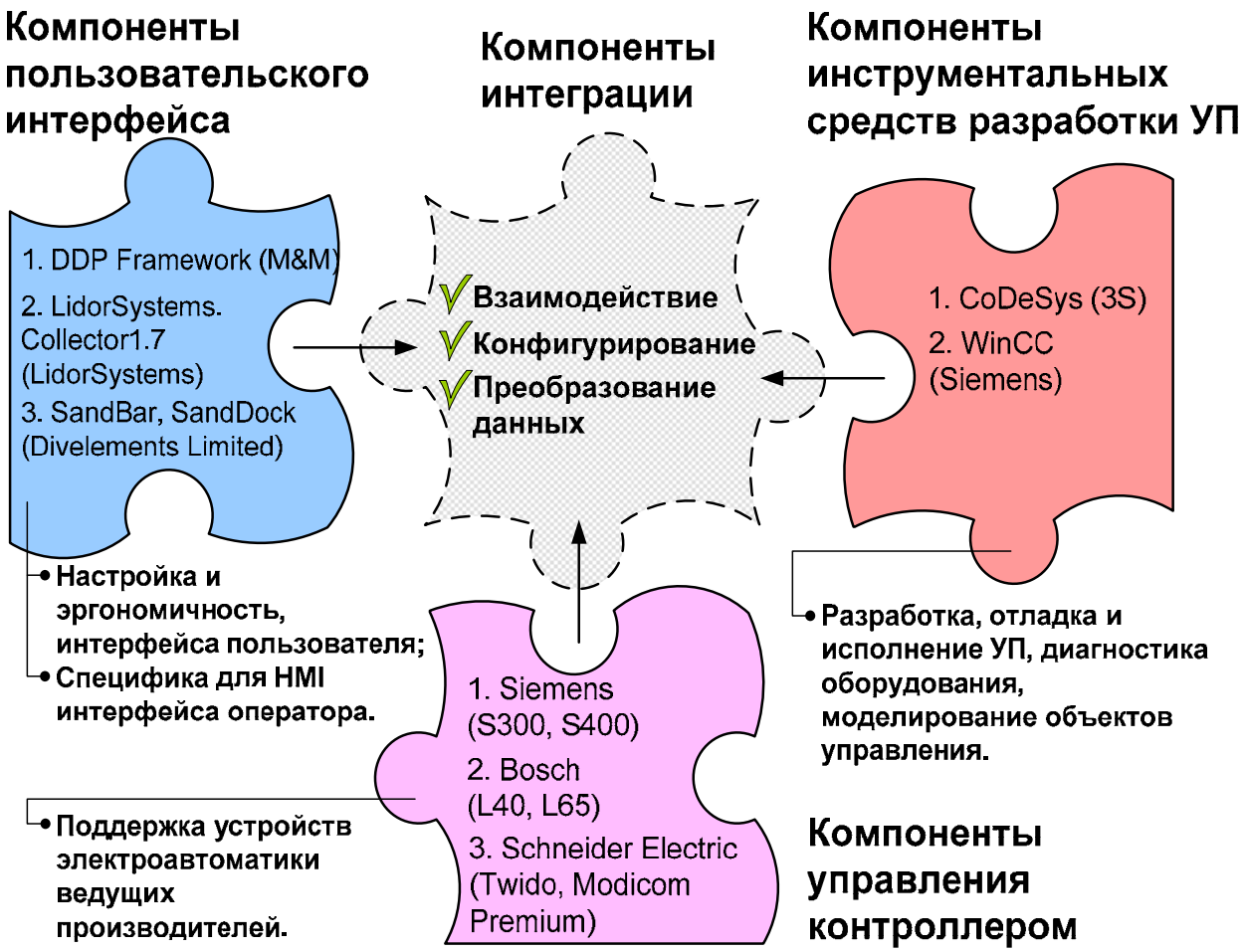

Рис. 1 Обобщенная архитектура открытого прикладного программного обеспечения электроавтоматики

Исследованы возможность и преимущества использования готовых проблемно ориентированных компонентов сторонних производителей при

разработке прикладных приложений. Выявлена необходимость систематизации прикладных решений и наборов компонентов, а также необходимость сопоставления задач управления с компонентами реализующими их. Обоснована идея синтеза прикладных приложений из готовых компонентов и решений.

Предложена архитектура открытого прикладного программного обеспечения электроавтоматики (Рис. 1) позволяющая выбирать лучшие решения для любой её составляющей. Центральное место в архитектуре занимают компоненты реализующие интеграцию, т.е. взаимодействие и обмен данными между компонентами различных групп, как собственных, так и сторонних разработчиков.

В исследовании выявлено, что использование предложенной открытой архитектуры прикладной составляющей имеет ряд преимуществ, для разработчиков и конечных пользователей:

- Снижаются затраты, на разработку пользовательского интерфейса и прикладных приложений (например, редакторов УП, конфигураторов сетевых подключений и т.п.);
- Обеспечивается возможность определения необходимого набора компонентов, и реализуемых ими задач, на ранних стадиях проектирования прикладного программного обеспечения электроавтоматики;
- Сокращается время выпуска новой системы, за счет возможности выпуска облегчённой (предварительной) версий с последующим наращиванием функциональности.

На основе проведённого анализа и выявленных проблем, поставлена цель и сформулированы задачи исследования.

Во второй главе сформулирован принцип классификации компонентов прикладной составляющей электроавтоматики, заключающийся в построение матрицы компонентов.

Условно выделены две группы компонентов: первая – компоненты окружения – привязана к аппаратной платформе исполнения и операционной системе; вторая – прикладные компоненты – реализует непосредственно задачи системы управления электроавтоматикой.

Компоненты окружения представляют общую для всех компонентов функциональность: конфигурирование, взаимодействие с платформой исполнения, реализация элементов пользовательского интерфейса. Компоненты прикладной области независимы от платформы исполнения, они специализированны под реализацию конкретных пользовательских приложений. Предложенная классификация (см. Рис. 2) позволяет систематизировать компоненты электроавтоматики по группам и определять функциональную нагрузку, которую несет каждая из групп и подгрупп.

Компоненты окружения			Компоненты прикладной области							
Компоненты каркаса системы	Компоненты конфигурирования системы	Компоненты интерфейса пользователя	Разработка управляющих программ	Отладка управляющих программ	Выполнение управляющих программ	Моделирование объекта управления	Диагностика оборудования	Контекстная помощь	Конфигурирование PLC	Прочие...
Операционная система			PLC - контроллер							

Рис. 2 Классификация компонентов систем управления электроавтоматикой

Компоненты окружения возможно разбить на три базовые группы: каркас, конфигурирование и интерфейс пользователя. Посредством компонентов каркаса снимается зависимость системы от рутинной работы по реализации пользовательского интерфейса и от взаимодействия с платформой исполнения. Это позволяет переносить решения с других операционных систем (например, Unix) изменяя только компоненты каркаса. Компоненты конфигурирования определяют, какие компоненты загружаются в текущей версии системы. Компоненты пользовательского интерфейса

реализуют специфичные для систем управления функции интерфейса оператора.

Произведена структуризация области пользовательских задач, посредством предложенной матрицы прикладных компонентов.

Логические уровни \ Пользовательские задачи системы управления	Разработка УП	Отладка УП	Выполнение УП	Моделирование объкта управления	Диагностика оборудования	Конфигурирование PLC
Визуальное представление	Редакторы ST, LD, FBD, др. языков	Отслеживание значений переменных; Работа с точками останова	Панели управления оператора	Редакторы графических объектов	Конфигуратор диагностических-кого вывода	Редактор сетевых соединений PLC
Управление данными	Компиляторы; Генераторы машинного кода для процессоров	Модель данных наблюдения	Команды управления процессом исполнения	Команды работы в графическом редакторе	Экспорт диагности-ческой информации	Сервис сетевых подключений
Хранение и предоставление данных	Компоненты хранение кодов УП; Языковая модель УП	Компоненты хранение отладочной информации	ОРС клиент процесса исполнения УП	Библиотека графических объектов	Диагностика во время управления объектом; БД диагности-ческой информации	Драйвера для сетевых протоколов ОРС, Profibus, CAN
Взаимодействие с аппаратурой	Сервер данных адресов переменных	Эмуляторы PLC и шлюзов сети	ОРС сервер процесса исполнения УП	Клиент данных для визуализации	ОРС сервер диагностики	

Рис. 3 Принцип построения матрицы компонентов

Матрица компонентов строится в виде пересечения пользовательских задач по горизонтали с логическими уровнями виртуальной структуры по вертикали (Рис. 3). По аналогии с Windows DNA (Distributed interNet Application), в вертикальном сечение вводятся промежуточные слои бизнес-логики, для отделения компонентов представления в интерфейсе пользователя от компонентов, специализированных на работе с данными. Вводится дополнительный уровень - взаимодействия с аппаратурой, который будет реализовывать доступ к конкретным устройствам PLC. Таким образом, столбцы полученной матрицы определяют многообразие пользовательских задач системы управления электроавтоматикой, а строки - способ компонентой реализации (организации) программного обеспечения.

С использованием инструмента матрицы компонентов был произведён анализ систем управления электроавтоматикой ведущих производителей. В результате анализа компонентов выявлена избыточность для системы WinCC (что характерно для систем управления Siemens). Оптимальной модульной архитектурой обладает система CoDeSys, хотя обнаружена недостаточная реализация диагностикой задачи. В системе лучше всего соблюдено компонентное разграничение пользовательских задач по логическим уровням. Что позиционирует её как потенциальный базис для развития прикладного программного обеспечения электроавтоматики.

С целью своевременного выпуска на рынок облегчённой версии программного обеспечения без ожидания конца разработки полной версии продукта, необходимо определять оптимально необходимый набор прикладных компонентов уже на начальных стадиях разработки. Эту задачу предложено решать с помощью матрицы компонентов.

Третья глава, посвящена созданию методики проектирования и разработки прикладной составляющей электроавтоматики с применением готовых компонентов внешних производителей.

Схематично методика представлена как набор шагов (Рис. 4), итерационно выполняемых в ходе очередного цикла разработки программного обеспечения электроавтоматики.

Сначала в соответствии с решаемыми прикладными задачами определяем набор компонентов с помощью матрицы компонентов. Затем выстраиваем обобщенную архитектуру системы.

На втором шаге происходит формирование единого исполняемого окружения прикладных приложений, здесь осуществляется реализация компонентов конфигурирования и компонентов бизнес логики пользовательского интерфейса. В последствии их интерфейсы и типы предоставляются компонентам прикладной составляющей.

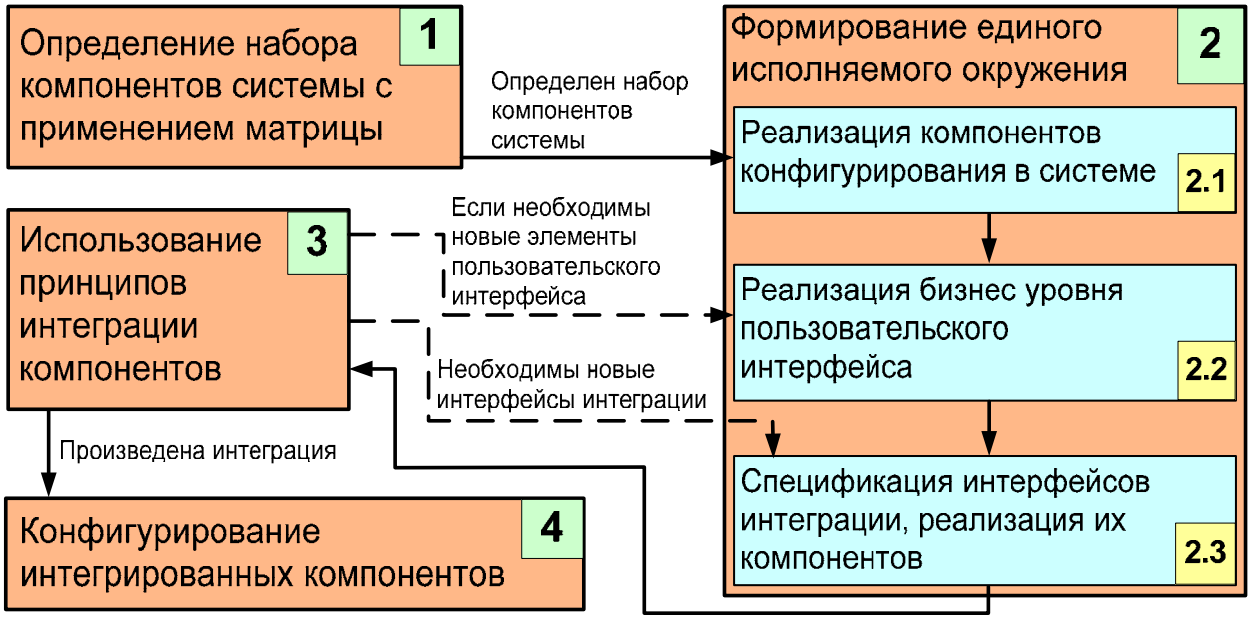

Рис. 4 Методика проектирования и разработки прикладной составляющей систем управления

Далее производится анализ и спецификация интерфейсов взаимодействия компонентов, определяются наборы интерфейсов, которые необходимы для реализации конкретных прикладных задач.

На третьем шаге производится интеграция компонентов в состав системы с применением специализированных интерфейсов и реализация связующего звена - компонентов интеграции.

В ходе интеграции выявляются новые требования к единому исполняемому окружению и необходимость реализации дополнительных интерфейсов бизнес уровня. В этом случае производится возврат на соответствующие этапы методики по формированию единого исполняемого окружения, для их последующего выполнения.

Одним из основных результатов применения единого исполняемого окружения и общего механизма интеграции компонентов является возможность конфигурирования компонентов системы, которое производится на последнем шаге.

Произведено сопоставление стадий Rational Unified Process (RUP), фаз процессов Microsoft Solutions Framework (MSF) с шагами описываемой

методики (Рис. 5). Это позволяет применять стандартные инструментальные средства в разработке прикладных приложений электроавтоматики.

Рис. 5 Исполнение шагов методики на стадиях RUP и фазах MSF для процесса разработки прикладной составляющей электроавтоматики

На первом шаге методики выполняются: выявление набора компонентов прикладной составляющей системы; определение необходимости в разработке новых или модификации существующих компонентов; систематизация требований и спецификация интерфейсов взаимодействия, типов данных, алгоритмов работы для новых компонентов.

Разработана последовательность деятельностей выполнения первого шага методики, с указанием исходных и получаемых данных, для определения требований к новым компонентам.

Сформулированы основные задачи, решаемые при создании единого исполняемого окружения: конфигурирование компонентов системы на базе общего механизма; реализация единого пользовательского интерфейса для прикладных приложений; отделение прикладных компонентов от специфики платформы исполнения, посредством уровня бизнес логики; предоставление общих интерфейсов интеграции для взаимодействия между компонентами.

Интерфейсы интеграции являются связующими элементами, определяющими правила взаимодействия, между компонентами системы. Интерфейсы интеграции разбиты на 3 группы в соответствии с их функциональным назначением. Специфицированы типичные интерфейсы интеграции для прикладных приложений электроавтоматики. Фрагменты таблицы с указанием номеров групп интерфейсов и их представителей приведены ниже (Таблица 1):

Таблица 1 Интерфейсы интеграции

Гр.	Интерфейсы	Реализуемая функциональность
1.	Интерфейсы каркаса прикладных приложений	
1.а	Интерфейсы конфигурирования: IConfig, IConfigManager	Определяют механизмы конфигурирования компонентов.
1.б	Интерфейсы базовых функций пользовательского интерфейса: IControlPanel, IViewPanel, IEditorPanel, IControlPanelsManager	Обеспечивает работу с панелями управления (меню, тулбары, дерево навигации и т.п.). Посредством этих интерфейсов создаются и размещаются элементы управления пользовательского интерфейса.
2.	Интерфейсы окружения	
2.а	Интерфейсы специализирован-ных функций пользовательского интерфейса: IPrintManager, IOptionManager, IWaitManager, IStatusManager	Предоставляют прикладным компонентам доступ к общим механизмам работы в пользовательском интерфейсе. Таким как, вывод на печать, предоставление странички настроек, отображение диалога ожидания или информации в троке состояния.
2.б	Интерфейсы ведения проекта: IPojectManager, ProjectEntry	Предоставляет функции для работы с проектом системы управления.
2.в	Интерфейсы администрирования системы и управления доступом: IContextHelpManager, IHelpPublication, IUserManagement, IPermissionChecker, IUserGroup, IUser	Обеспечивают функции помощи, и управления правами пользователей для работы в системе.
3.	Интерфейсы взаимодействия прикладных компонентов	
3.а	Интерфейсы взаимодействия с аппаратурой электроавтоматики: IGatewayClient, IOnlineDevice ISCPClient, IUIOnlineMode, IOnlineUIServices	Предоставляют механизмы работы с аппаратными средствами и контекстами состояний при подключении к PLC устройствам.
3.б	Интерфейсы обмена данными прикладных задач: ICompiledObject, ICompileContext,	Организуют доступа к данным компонентов различных задач. Например, данных о переменных в управляющей программе, данные о

Гр.	Интерфейсы	Реализуемая функциональность
	ICodegenerator, ILanguageModelProvider	параметрах генератора машинного кода или языковой модели для управляющих программ.
3.в	Интерфейсы управления и обмена сообщениями прикладных задач: IBreakpoint, IBPManager, IMessage, IMessageStorage, ISourceControlBinding	Реализуют управление функциями и предоставляют события работы прикладных компонентов. Например, получение событий об срабатывании точек останова, управление выводом сообщений для пользователя и получение команд от подсистемы контроля изменений в исходных кодах управляющих программ.

В основе принципа интеграции, заложено использование программных интерфейсов единого исполняемого окружения. Сама задача интеграции сводится к связыванию посредством интерфейсов в единую систему модулей готовых программных решений. Связывающим звеном здесь выступает специальный модуль, разработанный для передачи и трансформации данных между внедряемыми модулями – это компоненты интеграции.

Создание компонентов интеграции соответствует декомпозиции трех базовых деятельностей основных этапов внедрения компонентов в состав системы: интеграция визуальных объектов, интеграция программных модулей, интеграция функциональности. Базовые деятельности разбиваются на составные элементы, которые можно сгруппировать по уровням, в соответствие с функциональностью внедряемых компонентов в составе системы (Рис. 6). Например, “Компонентный” уровень включает в себя деятельности, осуществляемые на начальных стадиях внедрения компонентов в систему.

Уровни интеграции	Группы деятельностей			
1 уровень: Компонентный	Реинжиниринг	Интеграция COM интерфейсов и серверных компонентов	Реализация межъязыковых шлюзов	Установка связей с контейнерными элементами
2 уровень: Пользовательского интерфейса	Эргономическая интеграция	Языковая локализация	Интеграция алгоритмов работы с программой пользователем	Интеграция функций помощи (Help)
3 уровень: Логический	Интеграция алгоритмическая	Синхронизация состояний	Определения путей взаимодействия с аппаратурой	
4 уровень: Аналитический	Комплексный анализ работоспособности	Автоматизированное тестирование		
5 уровень: Визуальный	Реализация визуальных элементов	Интеграция локальных визуальных настроек		
6 уровень: Контекстный	Интеграция файловой структры	Настройка окружения		

Рис. 6 Диаграмма деятельностей интеграции компонентов в нотации UML

Предложенный порядок уровней интеграции учитывает особенности разработки прикладных приложений электроавтоматики и допускает повторное использование в последующих этапах развития системы.

Формально конфигурирование прикладной составляющей электроавтоматики разделено на три последовательных этапа.

1. Этап конфигурирования режимов системы закрепляет набор прикладных задач электроавтоматики за конкретным режимом системы управления. Режимы формируются на базе пользовательских задач электроавтоматики, при этом одна и та же задача может использоваться в нескольких режимах одновременно.

2. Этап конфигурирования задач определяет компоненты, которые используются в каждой пользовательской задаче. Задачи, одновременно используемые в разных режимах, могут иметь разную конфигурацию. Например, режимы программирования и отладки использует задачу моделирования объекта управления. Разница в том, что режим отладки не

использует компоненты для редактирования модели объекта управления и для работы с библиотеками визуальных объектов.

3. Этап конфигурирования интерфейса пользователя предполагает настройку компонентов уровня визуального представления, реализующих различные задачи, и размещение их элементов управления в окнах прикладных приложений.

Предложенная организация деятельностей анализа, проектирования и разработки прикладного программного обеспечения электроавтоматики, определяет метод создания прикладных приложений электроавтоматики на основе готовых решений внешних производителей. Согласно этому методу на этапе анализа используются инструменты реинжиниринга, в построении моделей для создаваемых или модифицируемых приложений, в том числе осуществляется классификации компонентов прикладных приложений. Этапы проектирования и разработки систем управления в коллективном исполнении требуют применения инструментальных средств ведения проекта, управления требованиями, построения моделей организации компонентов, а также инструментов управления изменениями исходных кодов, базы данных дефектов и средств автоматизированного тестирования.

В четвёртой главе, на базе предложенных теоретических и практических результатов решены три разных типа прикладных задач электроавтоматики. Предложено расширение функций системы CoDeSys в области трёхмерного моделирования объекта управления, управления доступом пользователей и конфигурирования системы.

В соответствии с методикой в начале разработки производился анализ компонентов системы CoDeSys, в результате было сформировано представление о структуре и механизмах взаимодействия прикладных компонентов. В системе разграничение специализации компонентов достигается применением механизма плагинов (съёмных компонентов) в архитектуре системы.

На основе проведённого анализа единого исполняемого окружения CoDeSys выявлено что, для реализуемых прикладных задач моделирования объекта управления и защиты авторских прав (управления доступом пользователей) необходима реализация компонентов системы в виде плагинов. Для этой цели в системе были выделены ряд интерфейсов интеграции из группы каркаса приложений (см. группу 1 в Таблица 1).

Решение задачи трехмерной визуализации объекта управления. Заложенная возможность двумерной графической симуляции объекта управления в CoDeSys не всегда соответствует современным требованиям сложных технологических процессов. Графический редактор 3D-сцены визуализации предоставляет пользователю средства трех мерного представления объекта управления и технологического процесса. Исполнительные звенья механизмов, их приводы и объекты окружения сцены размещаются и компонуются пользователем при построении модели технологического окружения управляемого процесса.

Элементы для построения 3D-сцены, такие как промышленные роботы, автоматические линии, захваты, манипуляторы, объекты технологического окружения и т.п., разрабатываются с использованием инструментария 3D Studio Max (фирмы Autodesk Media & Entertainment). Функции представления этих объектов в сцене осуществляет компонент библиотеки визуальных объектов (Рис. 7). Оконный элемент библиотеки используется для выбора и перетаскивания из него необходимых объектов при визуальной разработке моделируемого оборудования.

В начале внедрения компонентов был построен фрагмент матрицы, для систематизации компонентов графического редактора и компонентов интеграции. Далее выделены интерфейсы интеграции. Классы редактора и его инструментария были вынесены в компоненты плагинов CoDeSys. В результате, в систему был внедрён редактор сцены визуализации и его инструментарий (Рис. 7).

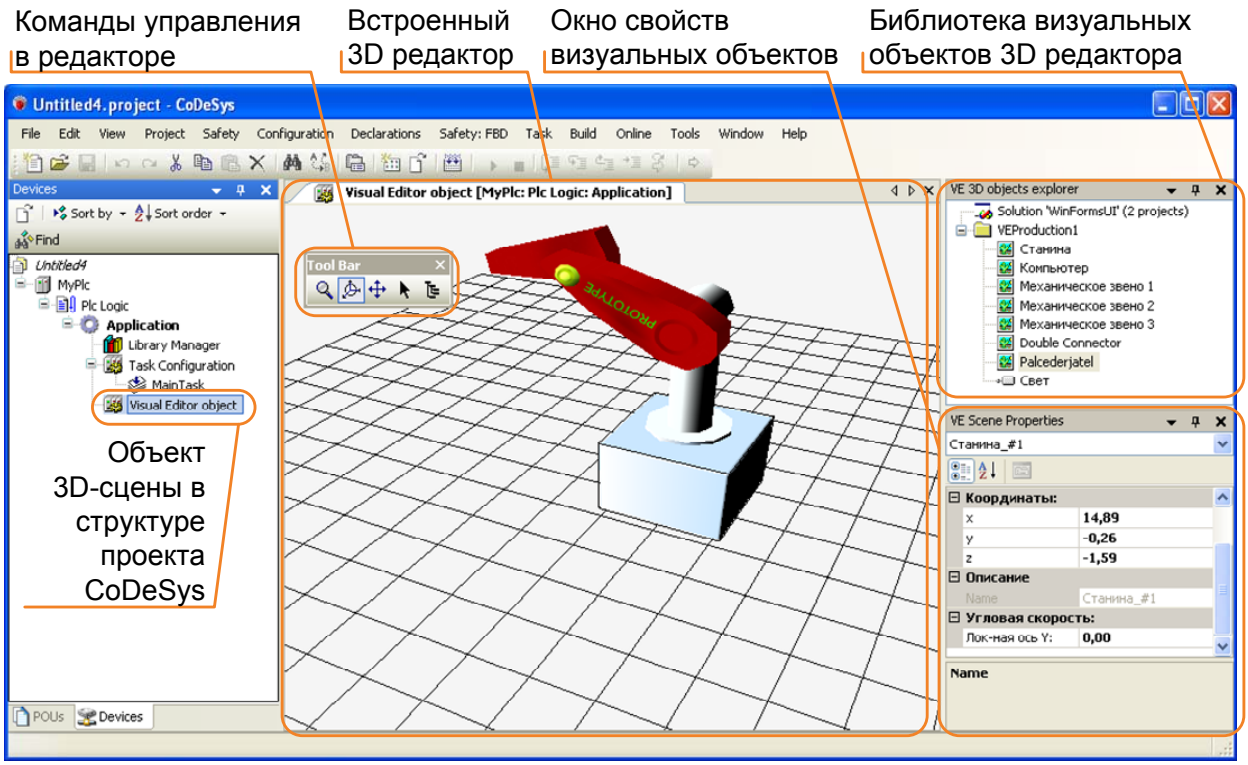

Рис. 7 Элементы управления редактора 3D-сцены в CoDeSys

Решение задачи управления доступом. При создании объектов управляющих программ применяют алгоритмы, являющиеся интеллектуальной собственностью разработчика или использующие не документированные способы взаимодействия с аппаратными средствами электроавтоматики, которые также не желательны для распространения в исходном коде. Предложено решение для защиты прав авторства, дополняющий имеющийся механизм управления правами пользователей в CoDeSys. С помощью этого решения разработчик предоставляет конечному пользователю заданную функциональность управляющей программы, но скрывает от его глаз код реализации. Таким образом, он сохраняет свое “know how”.

Для разработки и встраивания компонентов защиты в соответствии с классификацией была определена специализация разрабатываемых компонентов, определены необходимые интерфейсы интеграции (Рис. 8).

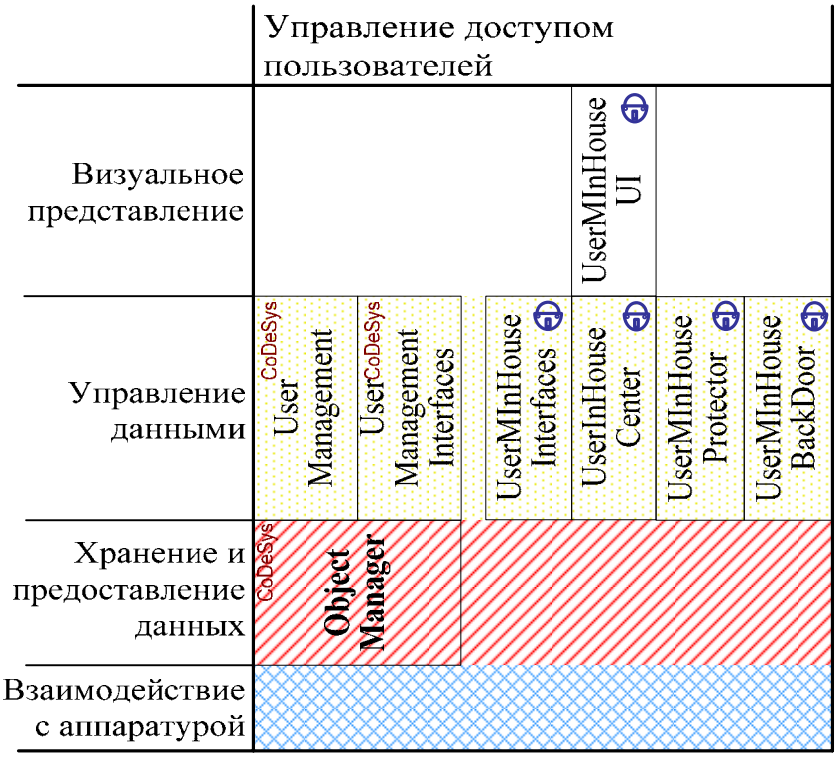

Рис. 8 Специализация разрабатываемых компонентов управления доступом в задаче управления доступом системы CoDeSys

На базе интерфейсов были реализованы команды контекстного меню в дереве проекта для защиты объектов управляющих программ от несанкционированного просмотра и редактирования. Разработанные компоненты (на Рис. 8 помечены замочком) производят доступ к объектам проекта управляющих программ посредством интерфейсов менеджера объектов системы CoDeSys. Это позволяет модифицировать данные в объектах для ограничения доступа пользователей при их использовании.

Решение задач конфигурирования, потребовало разработку дополнительных функций в утилите создания и изменения профилей использования прикладных компонентов CoDeSys. Стандартная конфигурация системы имеет 121 прикладной компонент. Для определения компонентов, которые используются в конкретной пользовательской задаче – второй этап конфигурирования, реализованы новые функции в утилите конфигурирования.

Пользовательский интерфейс утилиты представлен на рисунке (Рис. 9). Имена компонентов подсвечиваются цветом реализуемой ими задачи.

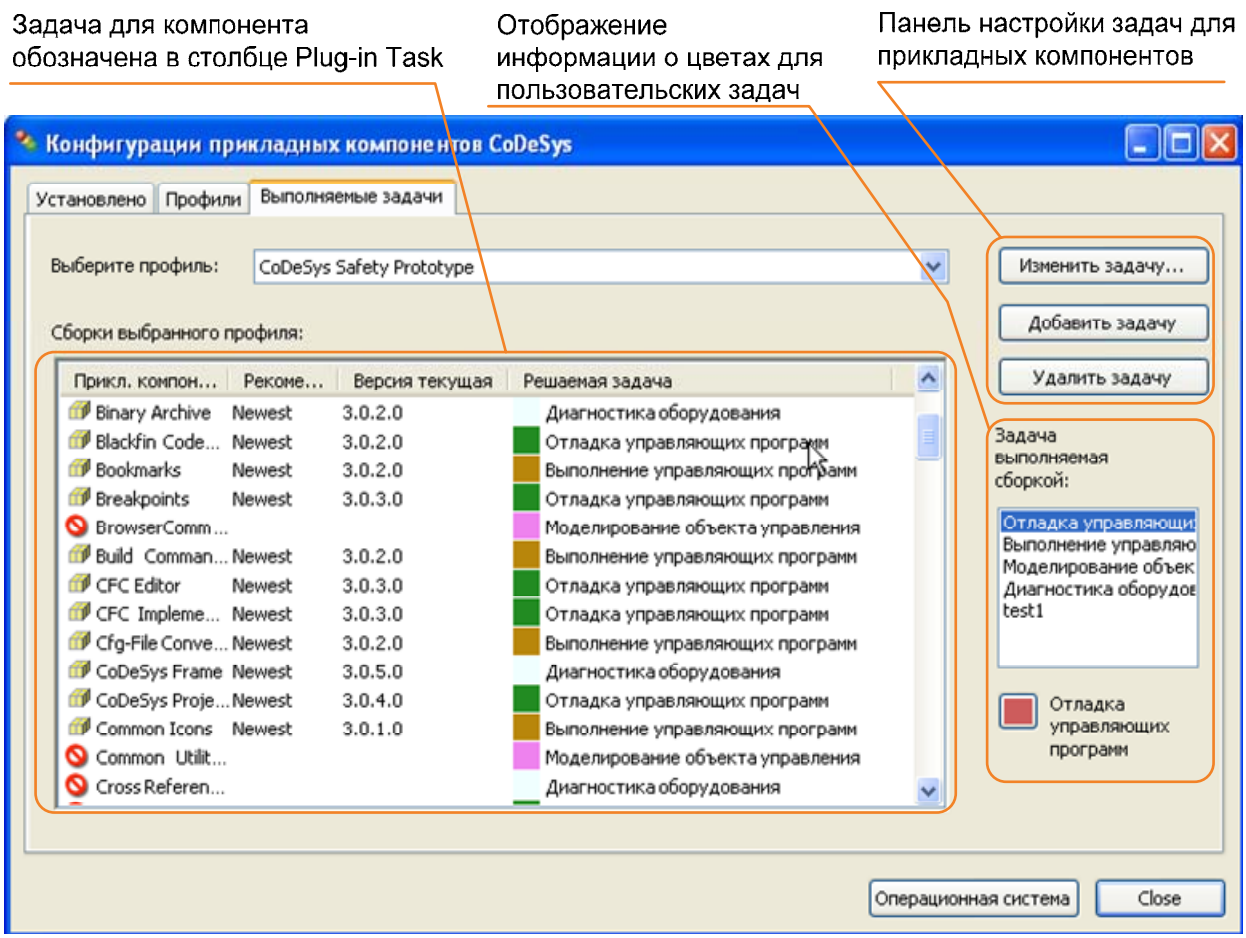

Рис. 9 Настройка конфигурации CoDeSys с применением принципов классификации прикладных компонентов

Реализация третьего шага - настройка расположения элементов управления в рабочей области экрана - потребовала реализацию двух новых компонентов. Это утилита разработки шаблонов расположения оконных элементов ToolWindowsLayoutSetting и компонент наблюдатель LayoutModeObserver, осуществляющий изменение расположения элементов пользовательского интерфейса при переходе системы из режима в режим.

Принцип взаимодействия внедряемых компонентов с компонентами окружения представлен на диаграмме (Рис. 10). В ToolWindowsLayoutSetting, создаётся набор файлов в xml формате, в которых хранятся данные о расположении оконных элементов в различных режимах. Для получения данных этих файлов используются компоненты пользовательского интерфейса SandDock, применяемые в CoDeSys.

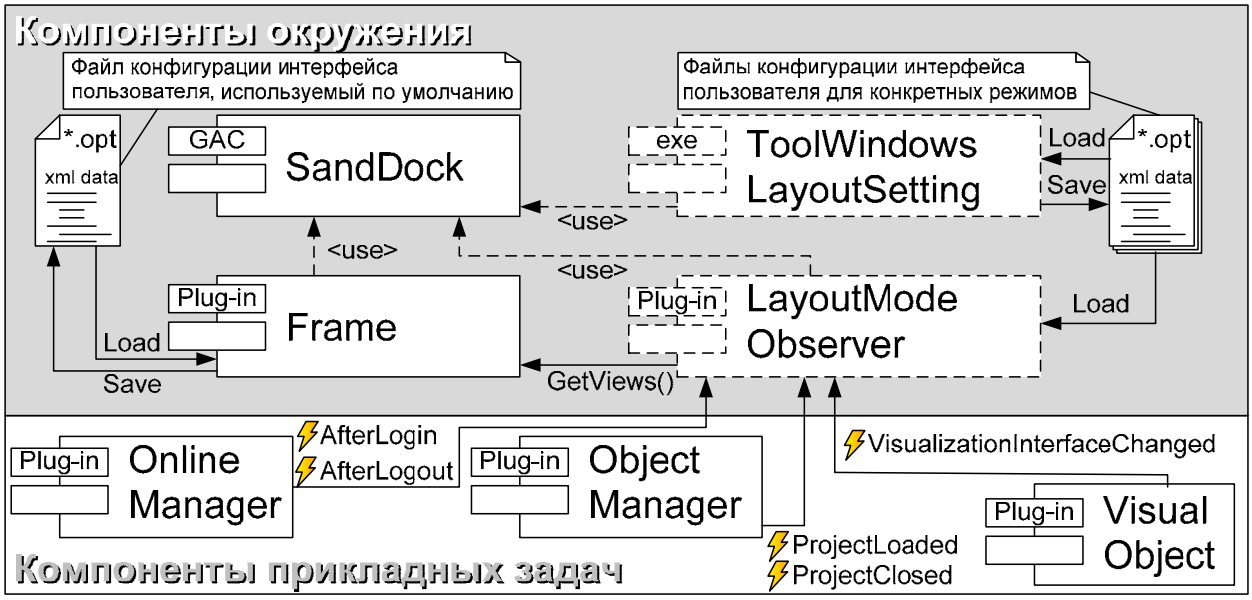

Рис. 10 Принцип работы компонентов по настройке интерфейса пользователя

Компонент LayoutModeObserver обрабатывает случаи изменения режима системы, посредством обработки событий от различных прикладных компонентов. Например, компонент OnlineManager, инициирует события о переходе в режим отладки, менеджер объектов ObjectManager предоставляет события открытия и закрытия проекта системы, а компонент объекта визуальной сцены VisualObject инициирует событие перехода в режим моделирования. При переходе в очередной режим, LayoutModeObserver загружает данные о размещении оконных элементов из xml файлов и перераспределяет оконные элементы в рамках рабочего экрана системы с использованием менеджера размещения в SandDock.

ОСНОВНЫЕ ВЫВОДЫ И РЕЗУЛЬТАТЫ РАБОТЫ

1. Решена задача повышения эффективности (снижение затрат и сокращение времени разработки, при сохранении того же качества) процесса разработки прикладного программного обеспечения промышленной электроавтоматики на основе его формализации, создания единого информационного окружения выполнения и обеспечения интеграции внешних программных компонентов.

2. Установлены взаимосвязи в архитектуре прикладной составляющей промышленной электроавтоматики на основе которых предложена

классификация с помощью матрицы компонентов, позволяющая систематизировать программное обеспечение по прикладным задачам (разделение по горизонтали) и оценивать модульность архитектурных решений (разделение по вертикали) с точки зрения адаптации программных компонентов к конкретным прикладным решениям и прогнозировать последующие эволюции системы.

3. Предложенный метод проектирования и разработки прикладной составляющей систем управления электроавтоматикой позволяет формализовать процесс создания приложений электроавтоматики, выявляет набор требуемых компонентов, обеспечивает внедрение решений внешних производителей, организует взаимодействие программных компонентов в едином исполняемом окружении.

4. Минимальный набор прикладных задач, достаточный для опережающего выпуска на рынок облегченной версии программного обеспечения, без ожидания конца разработки продукта, рекомендуется определять использованием разработанного инструмента матрицы компонентов.

5. Конфигурирование прикладных приложений под конкретные производственные потребности и настройка интерфейса пользователя по запросам оператора достигается применением предложенного компонентного подхода к организации программного обеспечения электроавтоматики.

6. Применение разработанного принципа интеграции внешних компонентов и предложенного единого исполняемого окружения позволяют, используя лучшие программные решения из смежных областей, повысить эффективность разработки и предоставлять новые пользовательские функции в прикладной составляющей промышленной электроавтоматики.

7. Задачи трехмерной визуализации объекта управления, администрирования прав доступа и конфигурирования прикладных приложений электроавтоматики рекомендуется решать с использованием

созданного метода проектирования и разработки прикладных приложений электроавтоматики.

8. Полученные теоретические и практические результаты применяются в курсах лекций и лабораторных работах учебных дисциплин “Структура и математическое обеспечение систем управления” и “Графические системы и интерфейсы оператора” на кафедре “Компьютерные системы управления” ГОУ ВПО МГТУ “Станкин”.

СПИСОК РАБОТ, ОПУБЛИКОВАННЫХ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Мартинов Г. М., Козак Н. В. Принципы интеграции компонентов электроавтоматики на примере 3D-сцены визуализации // Системы управления и информационные технологии . 2007. №2 (28). С. 88-92.
2. Мартинов Г. М., Козак Н. В. Декомпозиция и синтез программных компонентов электроавтоматики // Приборы и системы. Управление, контроль, диагностика. 2006. №12. С. 4-11.
3. Козак Н.В. Среда интегрированного функционирования в реализации интерфейса оператора ЧПУ // Труды Международной научно-технической конференции «Информационные средства и технологии». Москва 2004. - Том 3 – С. 150 – 153.
4. Козак Н.В., Бабак Д.А., Тихомиров В.В. 3D визуализация работы программ электроавтоматики // Труды Международной научно-технической конференции «Информационные средства и технологии». Москва 2005. - Том 3 – С. 11 – 13.
5. Козак Н.В. Классификация программных компонентов электроавтоматики // Труды Международной научно-технической конференции «Информационные средства и технологии». Москва 2005. - Том 3 – С. 26 – 29.
6. Козак Н.В. Киселёв С.А. Моделирование работы управляющих программ контроллеров при помощи 3D моделей // Труды Международной научно-технической конференции «Информационные средства и технологии». Москва 2006. - Том 3. – С. 168 – 171.
7. Козак Н.В. Мартинов Г.М. Декомпозиция программных компонентов электроавтоматики // Труды Международной научно-технической конференции «Информационные средства и технологии». Москва 2006. - Том 3 – С. 181 – 184.
8. Козак Н.В. Основы интеграции компонентов электроавтоматики на примере графического редактора 3D сцены визуализации // Труды Международной научно-технической конференции «Информационные средства и технологии». Москва 2007. - Том 3 – С. 132 – 135.