Содержание

Список используемых сокращений

АСУ ТП — Автоматизированная система управления технологическим процессом

ППВМ — Программируемая пользователем вентильная матрица

ЧПУ — Числовое программное управление

ERP — Enterprise resource planning

FPGA — Field-programmable gate array

IIRA — Industrial Internet Reference Architecture

MES — Manufacturing execution system

RAMI — Reference Architectural Model Industrie 4.0

SCADA — Supervisory control and data acquisition

SOA — Service-oriented architecture

XML — eXtensible markup language

WSDL — Web Services Description Language

Введение

Анализируя тенденции современного производства, стоит отметить одно наиболее важное изменение — переход от массового производства конвейерного типа к гибким автоматизированным и роботизированным производственным комплексам. Начало распространения комплексной автоматизации было обусловлено множеством технических предпосылок, и в первую очередь – это появление в начале 80-х годов достаточно мощных микропроцессоров и носителей памяти, а также удешевление и распространение компьютерной техники, что позволяло разрабатывать и внедрять такие системы. Естественно, одним из направлений автоматизации являлась автоматизация технологического оборудования. Благодаря научно-техническому прогрессу в этой области, удалось создать точное высококлассное оборудование с числовым программным управлением (ЧПУ) и привести его к такому виду, каким оно является сейчас.

В настоящее время информационные технологии распространяются повсеместно, и данный процесс не обходит стороной и производственную сферу. Фактически, промышленность вступает в новую эпоху Четвертой промышленной революции (известной также как Индустрия 4.0[1]), которая влечет за собой массовое внедрение киберфизических систем на производстве. на предприятиях присутствует огромное количество задач разных уровней, решение которых должно быть автоматизировано. Для этого применяется совокупность информационно-управляющих систем иерархически связанных друг с другом: ERP системы, MES, SCADA системы и встроенные системы полевого уровня. Однако не стоит забывать, что технологическое оборудование, являясь одной из важней-

[1]Индустрия 4.0 — это понятие, введенное в Германии в 2011 году, обозначает собой государственную программу поддержки и развития промышленности, главной стратегией которой является интеграция киберфизических систем в производственные процессы.

ших частей производства в целом, представляет собой не менее сложную информационно-управляющую систему. и облегчение процесса внедрения оборудования в общую производственную сеть является одной из главных задач Индустрии 4.0.

На данный момент в России уже предпринимаются попытки решения данной проблемы, так, например, разрабатывается и применяется платформа промышленного интернета Winnum[2] [1]. Она позволяет считывать и обрабатывать данные с разнообразных станков и подсоединять их к общей производственной сети с помощью дополнительного узла. Интеграция в сеть с использованием такого узла требует трудоемких настроек, что обусловлено существенными различиями в организации систем управления производителями технологического оборудования, однако является приемлемой как временная мера на этапе перехода.

В наши дни система управления технологическим оборудованием представляет собой монолитную систему, части которой жестко интегрированы и неотъемлемы друг от друга. Все это вызывает необходимость создания дополнительного слоя управления над таким оборудованием для объединения его с современной киберфизической системой, что в свою очередь требует значительных финансовых и временных затрат. Очевидно, что необходим пересмотр самой парадигмы проектирования оборудования с ЧПУ. Нужно рассматривать любое новое оборудование с точки зрения возможности включения его в единую информационно-телекоммуникационную среду с использование открытого протокола.

Все вышеперечисленное указывает на **актуальность** рассматриваемой темы и доказывает необходимость как пересмотра парадигмы построения ЧПУ систем в целом, так и компонентов системы управления, в том числе и человеко-машинного интерфейса.

[2]Winnum — это платформа промышленного интернета, главными задачами которой являются мониторинг, диагностика и оптимизация производственных процессов и оборудования.

Научная новизна выполненной работы заключается в предложенном структурно - модульном подходе к разработке графических интерфейсов модульной системы управления технологическим оборудованием.

Практическая ценность работы заключается в следующем:

– Разработаны программные модули компонента графического интерфейса и библиотека графических элементов системы управления технологическим оборудованием.

– Разработанный программный компонент был развернут на аппаратной платформе ODROID-C2[3].

– Налажено взаимодействие с ядром ЧПУ модульной системы управления технологическим оборудованием.

Объектом исследования является пользовательский интерфейс системы управления технологическим оборудованием. В качестве **предмета исследования** рассматриваются методики и инструменты, применяемые для его создания.

Целью данной работы является обоснование применения структурно-модульного подхода к разработке графического интерфейса системы управления технологическим оборудованием. Для этого требуется решить следующие основные **задачи**:

– Рассмотреть существующие методы и инструменты создания графического пользовательского интерфейса.

– Определить, какие требования к данному компоненту предъявляет модульная система управления, а также определить роль, которую занимает данный компонент, в соответствии с выбранной архитектурой системы управления.

[3]ODROID-C2 – 64-битный микрокомпьютер на базе ARM S905 Amlogic [2]

– Создать компонент пользовательского интерфейса и библиотеку базовых графических элементов.

Структура магистерской диссертации приведена далее. Магистерская диссертация состоит из введения, трёх глав, заключения, библиографии и трёх приложений.

Во введении рассматриваются актуальность темы работы, степень её исследования, а также объект и предмет исследования. Также сформулированы цель и задачи работы.

В первой главе приведен обзор состояния исследуемой темы, определены основные принципы создания модульной системы управления технологическим оборудованием, рассмотрена предлагаемая архитектура модульной системы управления.

Во второй главе определены основные функции пользовательского графического интерфейса в системах управления технологическим оборудованием, рассматриваются существующие методы и инструменты создания пользовательского графического интерфейса, проводится их сравнительный анализ. Обосновывается выбор протокола взаимодействия разрабатываемого модуля. Определяется роль и перечень требований к модулю графического интерфейса в реализуемой модульной системе управления технологическим оборудованием.

В третьей главе демонстрируются аспекты работы реализованного модуля пользовательского графического интерфейса.

В заключении приводятся результаты и выводы.

Глава 1

Обзор состояния вопроса и постановка задачи исследования

1.1 Степень изученности темы исследования

В последнее десятилетие стали активно проводиться работы по переходу к цифровому производству и интеллектуализации. Разработчики стремятся, чтобы вещи, которыми раньше управляли в ручную или с помощью жестких автоматических систем, получали доступ в сеть и сами договаривались о выполняемых работах. Технологическое оборудование является одной из ключевых и дорогостоящих частей производства, и для того, чтобы интегрировать его в обновленную производственную и технологическую среду, необходимо произвести обширные как научные, так и инженерные исследования.

Например, исследователи из университетов Керетаро (Autonomous University of Queretaro) и Гуанахуато (University of Guanajuato) в Мексике работают над созданием мультиагентной платформы для управления технологическим оборудованием [3]. Данная система называется MADCON[1] и представляет собой платформу с открытой архитектурой, основанную на мультиагентных программно-аппаратных модулях. Цель разработчиков – создание такой платформы, которая бы удовлетворяла требованиям реконфигурируемости для интеллектуальных машин следующего поколения. Аппаратные модули данной системы базируются на программируемых вентильных матрицах (ППВМ), также известных как FPGA, а программные модули используют XML для описания функций системы и графического интерфейса.

[1] MADCON — Multi-Agent Distributed Controller.

Аналогичные работы проводятся и в России. Например, ученые из Московского государственного технологический университет «СТАНКИН» предлагают подход к построению переносимого ядра ЧПУ на основе платформы независимых библиотек [4]. Открытая архитектура данной системы ЧПУ включает в себя уровни абстракции для реализации различных человеко-машинных интерфейсов, а также имеет возможность описания компонентов системы на различных языках программирования.

Компоненты системы связываются между собой по протоколам семейства Fieldbus[2], например, SERCOS [6, 7], EtherCAT [8], CAN-bus [9], Modbus [10] и т.д. Более того, специалисты университета самостоятельно разработали часть программно-аппаратных компонентов. Человеко-машинный интерфейс в данной системе обеспечивается тремя видами взаимодействия: с использованием физической стойки управления, удаленного устройства управления и веб-терминала, доступного с любого устройства с выходом в сеть.

В других странах также ведутся работы в этом направлении. В Хуажжунском университете науки и технологии (Huazhong University of Science and Technology) в Китае разрабатывают платформу для создания открытых ЧПУ систем [11]. Основными её задачами является упрощение и сопровождение разработки переиспользуемых модулей и интеграция их в прикладную систему управления. Кроме того, данная платформа включает инструменты моделирования и тестирования получаемых систем.

Кроме исследований, непосредственно связанных с созданием компонентов цифровых и интеллектуальных производств, ведутся работы по стандартизации. И лидирующую роль в этом вопросе занимают Германия и США. Немецкие специалисты разработали множество документов и стандартов, связанных с цифровым производством [12–14]. Основным

[2]Fieldbus – семейство протоколов помышленных сетей, используемых для распределенного контроля в реальном времени, описывается стандартом IEC 61158 [5].

является документ, посвященный эталонной архитектурной модели, также известной как RAMI 4.0 (Reference Architectural Model Industry 4.0) [15].

Параллельно в США были проведены аналогичные работы, однако стандарт был назван Industrial Internet Reference Architecture (IIRA) [16].

1.2 Основные принципы построения модульных систем управления технологическим оборудованием

Современные системы управления технологическим оборудованием отличаются комплексностью и монолитностью. Каждый их элемент – это «вещь в себе» с жёсткой и иерархической архитектурой. Всё в подобных системах направлено на обеспечение качества, надежности и бесперебойной работы. Инертность подобных систем заставляет разработчиков АСУ ТП следовать этому же принципу монолитности, потому что оборудование с централизованным управлением нельзя эффективно внедрить в децентрализованную производственную среду.

Вследствие этого, упор делается на интеграцию, то есть на объединение разрозненных компонентов в единую производственную систему, а не на интероперабельность – создание открытого интерфейса взаимодействия, позволяющего отдельным компонентам оставаться автономными, но способными общаться с другими компонентами в случае необходимости. Соответственно, желательно переходить к разработке оборудования с модульной архитектурой.

Подобная архитектура базируется на двух основных постулатах:

1. ***Унификация***. Под унификацией понимается открытая программно-аппаратная структура, позволяющая создавать новые типы оборудования и программного обеспечения по принципу «интеллектуаль-

ного конструктора». Унификация достигается за счет разбиения единого изделия на крупные взаимозаменяемые блоки с четким описанием входных и выходных параметров каждого блока.

2. ***Гибридизация***. Позволяет создавать установки, являющиеся совокупностью уже существующих. Например, можно сконфигурировать аддитивно-субстрактивную машину для быстрого прототипирования, сочетающую в себе характеристики 3D-принтера и трехкоординатного фрезерного станка или совместить фрезерную головку для черновой обработки заготовок с лазером для полирования определенных поверхностей.

Применяя данные принципы, в первую очередь необходимо выделить основные части технологического оборудования:

– рабочий орган;

– координатное шасси, перемещающее рабочий орган в пространстве;

– блок числового программного управления.

Очевидно, что координатное шасси является наиболее универсальным блоком, на который могут быть установлены различные рабочие органы, за счет чего может быть изменен тип оборудования. Например, сменив фрезерную головку на лазерную, можно сделать из фрезерного станка гравировальный или форматно-раскроечный, а поставив экструдер для пластика – 3D-принтер.

Однако не стоит забывать о том, что смена рабочего органа — это не только механическое изменение параметров оборудования, но и смена алгоритма управления. Блок числового программного управления должен заранее знать о каждом рабочем органе, который может быть установлен на шасси. Очевидно, что при таком подходе система остается монолитной

с иерархическим управлением. Поэтому необходимо выделить базовую часть архитектуры, определить спецификацию протокола взаимодействия и создания новых программно-аппаратных модулей, которые могут быть динамически включены в систему. К базовой части относится алгоритм управления электрическими приводами координатного шасси, а все остальное является внешними компонентами.

Подобный подход полностью изменяет стиль управления производственной средой, позволяя добиться бесшовного объединения различных производственных установок в единую сеть.

1.3 Микросервисная архитектура как основа создания модульных систем управления технологическим оборудованием

Сервис-ориентированная архитектура[3] является быстро развивающейся концепцией, все чаще используемой для реализации распределенных программных систем. Основа подобной архитектуры – набор слабо связанных заменяемых компонентов, оснащенных унифицированными интерфейсами для взаимодействия по стандартизированным протоколам. Первые упоминания о концепции SOA можно встретить в литературе начиная с 2009 года [17, 18].

Микросервисная архитектура[4] является современной интерпретацией SOA (первые упоминания об MSA относятся к 2012 году, хотя этот термин употреблялся и ранее), используемой для создания децентрализованного программного обеспечения [19]. В данной архитектуре под сервисом понимается процесс, выполняемый операционной системой, который взаимодействует с другими процессами по сетевому интерфейсу для

[3] В англоязычной литературе обозначается как Service-oriented architecture (SOA)

[4] В англоязычной литературе обозначается как Microservices architecture (MSA)

достижения общей цели. Микросервисы могут быть построены с применением любого языка или фреймворка, но должны использовать общий протокол, который позволяет скрыть особенности реализации каждого из микросервисов. При этом каждый микросервис работает лишь с небольшой, максимально ограниченной областью задач, выполняя минимум функций.

На сегодняшний день не существует четкого определения MSA. Тем не менее, из всего многообразия встречающихся в литературе формулировок попытаемся выделить те особенности данного подхода, которые особенно важны при проектировании распределенных систем ЧПУ:

1. *Архитектура микросервисов позволяет легко заменять модули, входящие в состав системы.* В качестве примера можно представить человеко-машинный интерфейс для взаимодействия с оборудованием. Классический интерфейс – это набор физических элементов управления: индикаторов, кнопок, переключателей и т.д. Подобный интерфейс очень удобен в массовом производстве, когда оборудование используется в составе конвейерной линии. Однако в условиях промышленной лаборатории (FabLab), подобный физический способ взаимодействия может быть избыточным. Использование микросервисного подхода дает возможность легко заменить физическое устройство управления на виртуальное, что позволит управлять оборудованием, например, через веб-интерфейс с любого устройства, подключенного к сети.

2. *Все модули организованы вокруг функций.* Архитектура микросервисов позволяет разделять функционал каждого блока. Рассмотрим в качестве примера фрезерную головку. Данный модуль состоит из физической части: двигатель, патрон, контроллер, датчики и логической – модуль управления. Физически обе части рас-

положены в едином блоке. Но с точки зрения архитектуры – это два разных сервиса. Один из них отвечает за низкоуровневые команды управления приводом головки, сбором данных от датчиков и т.д. А другой – это высокоуровневый интерфейс пользователя, который может включать в себя доступ к элементам управления, заданием технологических параметров, и даже включать в себя систему подготовки управляющих программ, нацеленных именно на фрезерную обработку. Каждый из них отвечает строго за свою функцию системы и ничего не знает о реализации других микросервисов.

3. *Каждый микросервис является эластичным, легко модифицируемым, но при этом законченным программным продуктом.* Данное утверждение подразумевает, что при разработке системы ЧПУ необходимо придерживаться принципа увеличения связности и уменьшения связанности. Это позволяет с одной стороны сфокусироваться на разработке и отладке каждого отдельного блока, а с другой – дает возможность упростить методику добавления и модификации функций системы в целом.

Микросервисная архитектура является современным и быстроразвивающимся направлением развития кибер-физических производственных систем и может быть успешно использована для создания распределенных систем ЧПУ.

1.4 Выводы к первой главе

1. Проведенные исследования показали, что предметная область активно развивается в направлении повсеместного внедрения информационных и коммуникационных технологий, что сопровождается уве-

личением автономности и независимости компонентов. Это влечет за собой изменение парадигмы управления как производством в целом, так и отдельно взятыми компонентами, в сторону модульности. А соответственно, данное высказывание справедливо и для систем управления технологическим оборудованием.

2. Основной проблемой систем управления технологическим оборудованием является монолитность их архитектуры. Для реализации современной системы управления технологическим оборудованием, требуется делать упор на интероперабельность – создание открытого интерфейса взаимодействия.

3. Проведя исследование микросервисного подхода, можно сделать вывод, что данный подход применим к системам ЧПУ, и соответствует тенденциям развития промышленности, обусловленным четвёртой промышленной революцией.

Основной задачей представленного исследования является установление необходимости и состоятельности структурно-модульного подхода для реализации пользовательского графического интерфейса микросервисной модульной системы управления технологическим оборудованием.

Глава 2

Пользовательский графический интерфейс систем управления технологическим оборудованием

2.1 Виды пользовательского интерфейса систем управления технологическим оборудованием и его основные функции

Любая система управления технологическим оборудованием состоит из трех основных частей:

- ядро системы управления;
- программируемый логический контроллер;
- человеко-машинный интерфейс.

Обычно человеко-машинный интерфейс в технологическом оборудование реализуется двумя способами: *аппаратным* и *программным*.

Классическая реализация аппаратной стойки управления включает в себя множество кнопок, регуляторов и других физических элементов (рис. 2.1). При этом все чаще стали получать распространение стойки, которые имеют в своем составе сенсорные экраны (рис. 2.2), позволяющие в разы сократить количество физических элементов управления, оставив только самые критически важные, например, кнопку аварийного останова.

Одним из главных недостатков таких стоек является то, что они располагаются непосредственно рядом с технологическим оборудованием и не обеспечивают возможность удаленного управления.

Рисунок 2.1 — Стойка FANUC Series 0i-TD

Рисунок 2.2 — Стойка KSE-TOUCH CNC

Многие производители технологического оборудования, пытаясь решить эту проблему, прилагают программное обеспечение, установив которое на персональный компьютер, появляется возможность осуществ-

лять удаленное управление. Однако с таким подходом сопряжен *ряд проблем.*

Во-первых, обычно поддерживается только ряд известных платформ, что вносит ограничения в организацию управления оборудованием.

Во-вторых, технологическое оборудование имеет долгий срок службы, а учитывая скорость развития информационных технологий и компьютерной техники, такое программное обеспечение придется регулярно обновлять. При этом процесс обновления так или иначе будет требовать определенного набора действий от пользователя, поскольку изменения будут проходить именно на используемой целевой машине.

В-третьих, отсутствие мобильности, поскольку оператор все также остается привязан к определенному рабочему месту.

Вместе с тем, учитывая тенденции развития современных производственных систем, важно не только обеспечить удаленное управление, но и определенный уровень абстракции. Это позволило бы незаметно для нижнего уровня осуществлять управление и наблюдение как с помощью оператора, так и с использованием другой программной системы более высокого порядка.

Сейчас ***функции пользовательского интерфейса*** обычно подразделяются на *пять групп* [20]:

1. *Операционные функции.* Они используются часто и помогают управлять технологическим оборудованием, а также показывают его состояние. Они занимаются отображением позиции, расстояния, подачи по каждой оси, скорости шпинделя, выполняемого блока программы и статуса. Кроме того, предусмотрены функции, помогающие работать с технологическим оборудованием, такие как переме-

щение в режимах Jog[1] и MDI[2], поиск программ, редактор программ и управление инструментами.

2. *Функции настройки параметров.* В системе ЧПУ имеются различные параметры для внутреннего использования, которые подразделяются на три вида:

   – параметры станка, которые используются для настройки станка, сервопривода, смещения инструмента, рабочей координаты;

   – параметры программы, которые должны быть установлены при редактировании программы обработки детали;

   – параметры настройки, которые используются, чтобы адаптировать машину к требованиям пользователя.

   Эти функции предоставляют интерфейс для установки, хранения и поиска параметров.

3. *Функции редактирования программы.* Эти функции позволяют редактировать и модифицировать программу обработки детали, которая представляет собой G-код.

4. *Функции контроля и сигнализации.* Система ЧПУ всегда уведомляет пользователя о состоянии и, при необходимости, запускает на исполнение важные задачи и информирует пользователя о результате. Функции сигнализации оповещают о сигнале тревоги, методе аварийного восстановления и т.д.

5. *Служебные функции.* Помимо других четырех основных функций, для оказания помощи пользователям предоставляется множество

[1] Jog – режим, который может быть использован для перемещения по заданной оси с заданной скоростью.

[2] MDI (Manual data input) – режим ввода команды (например, G-код) для прямого выполнения.

полезных функций. Например, осуществление сетевого взаимодействия и управление данными пользователя.

Графический интерфейс в первую очередь предоставляет широкий набор возможностей по визуализации процесса обработки. Визуализация процесса обработки является важной, поскольку дает возможность пользователю оборудования определить состояние и этап процесса обработки в реальном времени. В технологическом оборудовании визуализация обычно осуществляется несколькими способами:

– Экран, отображающий проделанный путь рабочей головки. Он позволяет визуально оценить получаемый результат. Особенно полезен в случаях, когда нет возможности осмотра происходящей обработки с камер или в реальности, например лазерная обработка или сварка.

– Экран с управляющей программой и выделенной текущей строкой, позволяет ответить на вопросы: «какое действие сейчас выполняется?» и «какая часть работ уже выполнена?».

– Набор экранов, отображающих текущие координаты рабочей головки, текущие настройки, инструмент, скорости и т.д.

– Экран видеонаблюдения. Применяется, когда оборудование оснащено камерами, обычно когда нет прямого доступа для наблюдения рабочей области.

– Экран с 3D-моделью выполняемого процесса в реальном времени.

Из всего вышесказанного следует, что графическая часть пользовательского интерфейса позволяет наиболее полно и наглядно обеспечивать связь технологического оборудования с пользователем, используя современные технологии, например, 3D-моделирование и работу с видео.

2.2 Инструменты разработки графического интерфейса

2.3 Структурно-модульный подход к разработке графического интерфейса

2.4 Способ взаимодействия пользовательского графического интерфейса

Микросервисная архитектура модульной системы управления предполагает применение общего облегченного протокола взаимодействия, позволяющего скрыть особенности реализации модулей. Так как модуль пользовательского графического интерфейса является частью децентрализованной гетерогенной сети, необходимо использовать «умные» приемники сообщений и «глупые каналы» передачи данных. Подобное решение дает возможность делать модули системы максимально независимыми и сфокусированными на решении одной конкретной задачи.

Очевидно, что при работе в сетях TCP/IP, наиболее очевидным способом передачи данных являются «сырые» сокеты. Через этот программный интерфейс можно передавать байтовый поток данных. Но в разрабатываемой системе все модули, в том числе и модуль пользовательского графического интерфейса, обмениваются уже структурированной информацией. Поэтому более правильным подходом будет использование разновидности пакетной передачи данных. Каждый пакет – есть некоторое сообщение, которое может быть передано от одного модуля к другому. Получив данное сообщение, принимающий модуль должен подтвердить получение (в случае, если его содержимое ему понятно и может быть обработано) либо отклонить сообщение.

Легко заметить, что подобные сообщения очень похожи на те, которые используются в таких протоколах как XML-RPC, SOAP, BPEL или WSDL, но с одним важным отличием. Все эти протоколы используются в крупных промышленных приложениях и явно не подходят для встраиваемых систем. А так как модуль графического интерфейса должен осуществлять взаимодействие и со встраиваемыми системами, то данный недостаток является существенным. Для создания легковесной, открытой и расширяемой системы управления необходимо минимизировать слой абстракции, максимально используя преимущества низкоуровневых сокетов в сочетании с возможностью осуществлять маршрутизацию сообщений.

Таким образом, наиболее подходящим способом взаимодействиями модулей проектируемой системы числового программного управления являются ***очереди сообщений***.

Очередь сообщений является асинхронным протоколом передачи данных, то есть отправитель и получатель сообщения не взаимодействуют друг с другом напрямую, а только через очередь.

Можно выделить следующие ***основные особенности очередей сообщений***, которые позволяют сделать вывод о том, что для проектируемой системы управления данный способ передачи данных является наиболее целесообразным:

– Очереди сообщений позволяют компонентам системы оставаться максимально независимыми друг от друга, исключая возможные взаимные блокировки, когда один из участников взаимодействия вынужден ожидать освобождения сетевых ресурсов, необходимых ему для передачи или приема данных.

– Очереди сообщений дают возможность экономить вычислительные ресурсы за счет отсутствия необходимости в сетевых буферах,

в которых хранятся еще не переданные или еще необработанные данные. По сути, очередь сообщений сама является универсальным сетевым буфером, не привязанным к какому-то конкретному узлу или процессу.

– Очереди сообщений легко масштабируются как по объему передаваемых данных, так и по пропускной способности.

– Очереди сообщений позволяют легко сглаживать пиковые нагрузки, возникающие в сети. Для проектируемой системы управления это особенно важно, так как в ней возможно смешение разных типов трафика. В частности, можно выделить как минимум высокоприоритетный трафик (данные от датчиков, команды управления) и низкоприоритетный трафик (загрузка исходного текста программы управления в контроллер, передача видео из зоны обработки и т.д.).

– Очереди сообщений позволяют существенно повысить отказоустойчивость системы, ведь сообщения остаются в очереди и могут быть обработаны даже в случае отказа отправившего их узла. Это можно продемонстрировать на примере передачи управления управляющей программы для оборудования с ЧПУ. Управляющая программа создается оператором в блоке интерфейса, а затем передается в блок исполнения в виде потока текстовых команд. Очевидно, что программа будет передана в виде нескольких сообщений, помещаемых в очередь, причем на максимально возможной скорости. Предположим, что какой-то момент блок интерфейса «зависает», данный факт фиксируется сторожевым таймером, который инициирует перезагрузку операционной системы. Однако во время перезагрузки блок исполнения продолжает считывать и обрабатывать сообщения из очереди, и для него процесс передачи данных не прерывается.

– Очереди сообщений гарантируют доставку сообщения, по крайней мере до тех пор, пока в сети функционирует хотя бы один узел, который может обработать их.

– Очереди сообщений не нарушают порядок передаваемых сообщений. Как правило, сообщения будут получены именно в том порядке, в котором они были отправлены.

2.5 Отображение модуля на RAMI 4.0

Индустрия 4.0 включает в себя множество аспектов, и для того, чтобы определить общее виденье этих аспектов и их связь, была создана архитектурная модель RAMI 4.0 (рис. 2.3). Данная модель является трехмерной и по трём осям отображает предметную область с точки зрения информационных технологий, жизненного цикла изделий и организации производства.

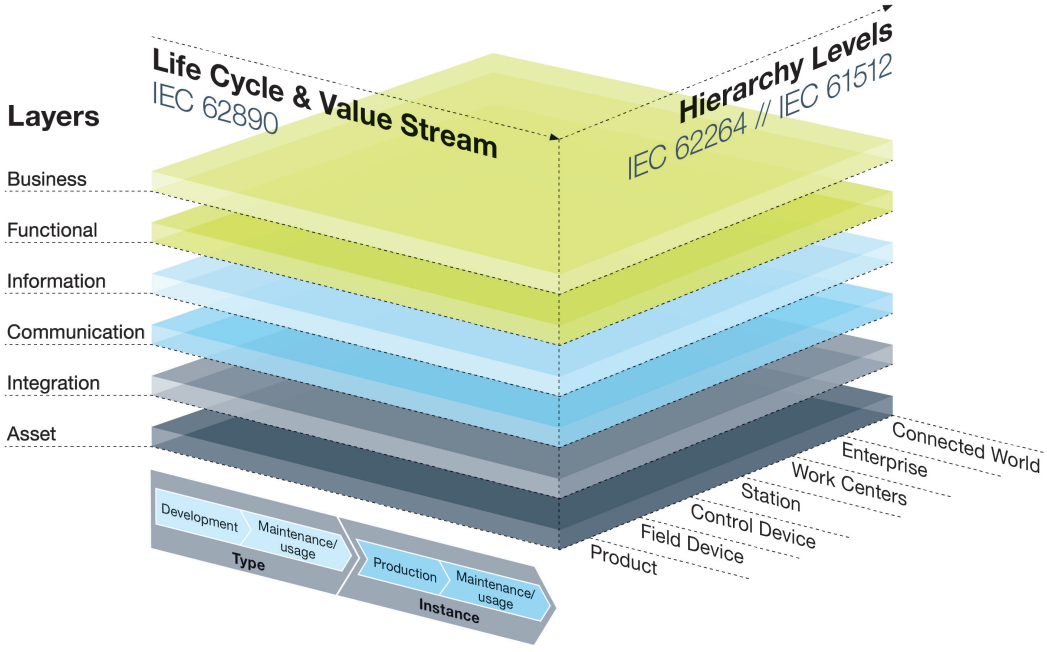

Рисунок 2.3 — Архитектурная модель RAMI 4.0

Одна из основных задач данной модели – обеспечить простое, наглядное и структурированное представление, которое бы помогало при созда-

нии описания компонентов и локализировало бы их роль в предметной области.

Таким образом, для того чтобы обосновать применение структурно-модульного подхода к реализации графических интерфейсов, далее будет представлено отображение на архитектурную модель RAMI 4.0 модуля пользовательского графического интерфейса, разработанного с использованием вышеуказанного подхода.

Модуль графического интерфейса входит в состав системы управления технологическим оборудованием, поэтому на правой горизонтальной оси он находится в областях *«устройство управления» и «станция»*. Модуль не входит в область «полевые устройства», поскольку не имеет связи с ними напрямую.

Наибольший интерес в данном случае представляет вертикальная ось. Модуль графического интерфейса находится на уровнях *«интеграция», «взаимодействие», «информация» и «функционал»*. Вместе с этим, он обеспечивает открытые связи с вышестоящим и нижестоящим уровнем.

Уровень интеграции. Цель данного уровня – преобразование событий реального мира в события виртуального мира. На этом уровне рассматриваемый модуль обеспечивает преобразование действий пользователя (нажатие на кнопки, ввод программы и т.д.) в события в виртуальном мире. При этом пользователь находится на нижестоящем уровне – уровне активов. Фактически, этим занимается клиентская часть рассматриваемого модуля – т.е. браузер и JavaScript обработчики событий, активирующиеся при взаимодействии пользователя с графическими формами.

С другой стороны, модуль графического интерфейса входит в распределенную гетерогенную сеть системы управления технологическим оборудованием. Поэтому преобразование происходит и в обратную сторону. Например, сигнал окончания обработки передается на модуль графического интерфейса, а тот в свою очередь извещает об этом пользователя.

Уровень взаимодействия. Целью уровня взаимодействия является определение способа коммуникации, т.е. интерфейса соединения, протокола и формата данных, а также обеспечение связи уровня интеграции и информационного уровня. Рассматриваемый модуль имеет несколько интерфейсов соединения – беспроводное WiFi соединение, Ethernet и последовательное соединение через UART. В качестве протокола используются очереди сообщений, реализованные с помощью библиотеки *nanomsg*. Очереди построены поверх разных протоколов, например, со стороны клиента поверх протокола websocket. В качестве формата передаваемых данных используется бинарный формат *cbor*.

Уровень информации. На данном уровне определяются данные, с которыми работает система, способы их хранения, описание событий и их предварительная обработка. В модуле графического интерфейса на этапе инициализации в качестве данных выступают статические файлы, которые передаются от других модулей системы управления. К статическим файлам относятся:

– html файлы, описывающие структуру графического интерфейса;

– css файлы, описывающие внешний вид графического интерфейса;

– js файлы, описывающие логику работы графического интерфейса.

Для хранения данных модуль графического интерфейса использует базу данных UnQLite [21]. Данная система является NoSQL[3] базой данных, основная особенность которой – отсутствие сервера. В отличие от таких известных NoSql движков как MongoDB[4], Redis[5], CouchDB[6], UnQLite

[3]NoSQL – подход к реализации хранилищ данных, где доступ к данным осуществляется без использования языка SQL.

[4]MongoDB – документо-ориентированная СУБД, использующая документы, основанные на формате JSON.

[5]Redis – сетевое хранилище типа «ключ-значение», осуществляющее хранение в оперативной памяти.

[6]CouchDB – документо-ориентированная СУБД, написанная на Erlang.

не требует установки или настройки, а также не запускает отдельного процесса для доступа к данным. Все данные хранятся в виде одного файла с сериализованными по стандарту JSON данными. В остальном, UnQLite является хранилищем пар «ключ-значение», с поддержкой курсоров и возможностью хранить данные как на диске, так и в оперативной памяти. В дополнение к этому, UnQLite может работать и как хранилище документов и поддерживает транзакции.

После этапа инициализации модуль работает с данными, вводимыми пользователями, например G-код программами, Gerber файлами и т.д.

Уровень функционала. На данном уровне определяется перечень функций, которые предлагаются для использования из вне, и платформа для их предоставления.

Фактически, сам по себе модуль графического интерфейса не обладает функционалом, доступным для других участников, за исключением удаленного доступа. Однако данный модуль является платформой, которую используют участники гетерогенной сети системы управления технологическим оборудованием для того, чтобы предоставить возможность воспользоваться своими функциями.

2.6 Выводы к второй главе

Глава 3

Реализация модуля графического интерфейса

3.1 Описание структуры модуля

3.2 Описание взаимодействия модуля

3.3 Выводы к третьей главе

Заключение

Список литературы

1. Winnum Platform [Электронный ресурс]. URL: http://www.winnum.ru (дата обращения: 12.04.2017).
2. User Manual for your ODROID-C2 [Электронный ресурс]. URL: http://odroid.com/dokuwiki/doku.php?id=en:odroid-c2 (дата обращения: 16.04.2017).
3. Morales-Velazquez L., de Jesus Romero-Troncoso R., Osornio-Rios R. A. et al. Open-architecture system based on a reconfigurable hardware-software multi-agent platform for CNC machines // Journal of Systems Architecture. 2010. Vol. 56, no. 9. P. 407 – 418.
4. Grigoriev S. N., Martinov G. M. Research and Development of a Cross-platform CNC Kernel for Multi-axis Machine Tool // Procedia CIRP. 2014. Vol. 14. P. 517–522. 6th CIRP International Conference on High Performance Cutting, HPC2014.
5. IEC 61158-1. Industrial communication networks - Fieldbus specifications - Part 1: Overview and guidance for the IEC 61158 and IEC 61784 series: Standard: International electrotechnical commission, 2014.
6. IEC 61800-7-1:2015. Adjustable speed electrical power drive systems - Part 7-1: Generic interface and use of profiles for power drive systems - Interface definition: Standard: International electrotechnical commission, 2015.
7. IEC 61784-1:2014. Industrial communication networks - Profiles - Part 1: Fieldbus profiles: Standard: International electrotechnical commission, 2014.
8. ISO 15745-4:2003. Industrial automation systems and integration – Open systems application integration framework – Part 4: Reference description for Ethernet-based control systems: Standard: International Organization for Standardization, 2003.

9. ISO 11898-1:2015. Road vehicles – Controller area network (CAN) – Part 1: Data link layer and physical signalling: Standard: International Organization for Standardization, 2015.
10. Modicon Modbus Protocol Reference Guide, 1996.
11. Bin L., Yun-fei Z., Xiao-qi T. A research on open CNC system based on architecture/component software reuse technology // Computers in Industry. 2004. Vol. 55, no. 1. P. 73 – 85.
12. IEC 62541-100:2015 OPC Unified Architecture - Part 100: Device Interface: Standard: International electrotechnical commission, 2015.
13. IEC 61987-11:2016 Industrial-process measurement and control - Data structures and elements in process equipment catalogues - Part 11: List of properties (LOPs) of measuring equipment for electronic data exchange - Generic structures: Standard: International electrotechnical commission, 2016.
14. DIN SPEC 40912:2014-11 Core models - Specification and Examples: Standard: German Institute for Standardization, 2014.
15. DIN SPEC 91345:2016-04 Reference Architecture Model Industrie 4.0 (RAMI4.0): Standard: German Institute for Standardization, 2016.
16. Lin S.-W., Miller B., Durand J. et al. The Industrial Internet of Things Volume G1: Reference Architecture: Tech. rep.: Industrial Internet Consortium, 2017.
17. Laskey K. B., Laskey K. Service oriented architecture // Wiley Interdisciplinary Reviews: Computational Statistics. 2009. Vol. 1, no. 1. P. 101–105.
18. SOA Manifesto [Электронный ресурс]. URL: www.soa-manifesto.org (дата обращения: 23.04.2017).
19. Microservices. A definition of this new architectural term. [Электронный ресурс]. URL: https://martinfowler.com/articles/microservices.html (дата обращения: 23.04.2017).

20. Suh S.-H., Kang S.-K., Chung D.-H., Stroud I. Theory and Design of CNC Systems. Springer-Verlag, 2008. 455 p.
21. An Embeddable NoSQL Database Engine. [Электронный ресурс]. URL: https://unqlite.org/index.html (дата обращения: 25.04.2017).

Приложение А

Имя приложения